



Virthos User Manual

Version 2.0
Revised August 25, 2006

© 2006

Virthos Systems GmbH

Contrescarpe 8c

28203 Bremen

Germany

Email: info@virthos.net

www.virthos.net

Virthos® is a registered trademark of
Virthos Systems GmbH, Bremen, Germany.

Table of Contents

Welcome to Virthos	7
About This Manual	9
Part 1: Basics	11
System Requirements	13
PHP Support	13
mySQL Support	14
Permissions	15
Space Requirements	15
Installation	16
Using the Online Installation Program	16
Manual Installation	17
Customizing the Installation	19
Special Cases	20
Concluding and Testing the Installation	21
Configuration	23
General Information About the Configuration File	23
Predefined Configuration Parameters	24
User Configuration Parameters	24
Licensing	25
Local Access	25
Remote Access	25
Ordering and Registering License Keys	26
Maintenance	27
Backing up Data	27
Restoring Data	28
Backup Strategies	28
Releasing Unused Disk Space	29
Restoring Default Settings	29
Domain and Server Replacement	29
Uninstalling	31

Part 2: Virthos Manager	33
Introduction	35
Starting and Stopping Virthos Manager	35
Virthos Manager's Regions	36
The Header Area	37
The Navigation Bar	37
The Mode Selector	39
The Toolbar	40
The Clipboard	41
The Trash Can	42
Managing Pages	43
Creating New Pages	43
Changing Page Content	44
Translating Page Content	45
Moving Pages	45
Copying Pages	46
Creating Links	47
Modifying Page Specifications	47
Changing the Sort Order	50
Removing Pages	51
Restoring Pages	51
Exporting Pages	51
Importing Pages	52
Editing Content	54
Formatting with Font Styles	54
Applying CSS Styles	55
Bulleting Text	55
Internet and Email Addresses	56
References to Other Virthos Pages	57
References to JavaScript Functions	58
Uploading Images and Files	59

Managing System Settings	61
General Settings.....	61
Database Settings.....	62
Email Settings	62
Licensing Settings	62
Security Settings.....	63
Language Settings	63
Custom Settings	65
Managing Access Rights.....	67
General Information About Access Rights	67
Access Rights in Virthos Basic.....	68
Access Rights in Virthos Pro.....	69
Managing Users	70
General Information About Users and Groups	70
Accessing User Management	71
Creating a New User	71
Creating a New Group.....	73
Changing User and Group Properties.....	74
Deleting Users and Groups.....	75
Part 3: Templates.....	77
Creating and Installing Templates	79
Design Rules.....	79
Naming Rules	81
Installing Templates	81
Post-Implementation Modifications.....	82
Template Variants	83
User Methods	83
System Methods.....	84
Global Methods.....	85
Language Variants	85
Introduction to VirthosTalk	86
General Syntax Rules.....	87
Editable Text.....	89

Interchangeable Images	90
Interchangeable Downloadable Files	92
Dynamic Lists	93
Conditional Layout Elements	95
Hyperlinks	97
Changing Context	99
Repetition	100
Variables	101
Library Files	102
URL Parameters	104
Form Data	105
Processing Forms	106
Basics of Working With Forms	106
Sending Form Data in Emails	108
A User-Designed Login Form	109
Creating New Pages Using a Form	110
Changing Page Content with a Form	111
Dynamic Navigation Menus	113
Troubleshooting	115
Part 4: Integration	117
Entry Addresses	119
General Observations About Address Parameters	119
References by Page Number	120
References by Path	120
Accessing Template Variants	121
Accessing Language Versions	122
Invalid References	122
Virthos as a Start Page	123
Frame-Based Layouts	123
Automatic Redirection	124
Reconfiguring a Web Server	125
Integrating Virthos With an Existing Website	126

Welcome to Virthos

We would like to thank you for your interest in Virthos, the content management system for websites of all kinds. With Virthos, you can create a presence on the Internet with very little effort, and maintain your site using only a browser. To use Virthos, a basic knowledge of HTML is the only requirement—you won't need to concern yourself with script programming or database administration—Virthos takes care of all that for you.

Virthos is available in two versions:

Virthos Basic	designed for smaller websites with a limited number of pages, maintainable by one person or a few persons. This version does not use a mySQL database, and can therefore be used on web servers that do not offer support for mySQL.
Virthos Pro	suitable for websites of all sizes. The underlying mySQL database lets you manage thousands of web pages. Since this usually amounts to too much work for one person or a few persons, Virthos Pro can support many users, each of whom has his own area of responsibility. Virthos Pro also lets you manage multilingual content.

If you have already worked with other content management systems, please note that Virthos takes a different approach from that of many of the other products that are currently on the market. Virthos is not based on either a design-oriented or topic-based structure. For example, it does not deal with “Articles”, “News” or “Discussion Groups” or anything of the sort—just pages with editable areas. The significance of a page—whether it is an article or a forum contribution—is determined by its form and the context in which it appears.

At first, not having a predefined structure may cause you some uneasiness because you have nothing to use as a starting point for your site design. On the other hand, you have a certain amount of artistic freedom not found in other solutions. The advantage for web designers who need to cater to the desires of different kinds of customers is that after each job, they do not need to select (or learn to use) a different platform, but can instead do almost anything with Virthos. Likewise, the advantage for customers is that when their needs evolve, they do not have to redo their site on a new platform, but can build on—or have a professional build on—their existing Virthos platform.

You can get some idea of how flexible Virthos is by considering some of the things for which it has already been used:

- “Latest Happenings” page
- Product catalogs and price lists
- News and events
- Media Center with image and file downloads
- Picture gallery
- Recipe page
- Online store with order form
- Contact form

- Registration forms
- E-learning application
- Discussion forum
- Guest book
- Download page
- Online database
- ...

Once you have mastered the fundamentals, you can put Virthos to work in many additional areas, even some that have only a minimal connection with web publishing. In a way, Virthos resembles a kind of operating system on which you can develop online applications of many sorts at little cost.

We hope that working with Virthos will bring you both pleasure and success!

About This Manual

This manual is intended for persons who want to build websites with Virthos. It assumes some degree of experience in designing and publishing pages on the Web; however, specialized knowledge of script programming and database administration are not required to understand the descriptions in this manual.

The manual is complemented by several other publications:

- the “Introduction to Virthos,” which provides an overview of the installation process, as well as a step-by-step introduction to Virthos' major functions.
- the “VirthosTalk Reference,” a complete description of the VirthosTalk language, used to develop scripts in Virthos.
- a variety of other publications, all available for download from www.virthos.net

In any case, before going through this manual, it would be a good idea to follow the procedures described under “Getting Started” in the introductory guide. You can also consult the VirthosTalk Reference from time to time if you need additional information for specific cases.

Virthos Basic / Virthos Pro

Virthos is available in two versions: Virthos Basic and Virthos Pro. This manual applies to both versions. Any material that applies to only one or the other version will be clearly marked as such.

Usage of Font Styles

Several different types of text are used in this manual in an effort to make clear how certain expressions are used.

- **Bold type** is used to indicate steps to be carried out in order to accomplish a task.
- **Monospace type** is used for text to be entered on the keyboard.
- *Italics* are used for text that appears on the screen, such as the names of files and directories
- the names of buttons that can be clicked appear in [square brackets].

Please note!

A box such as this indicates important instructions that you should not overlook. ¶

Presentation of Images

Illustrations of web pages and entry forms in this manual were made using the Safari web browser on a Mac OS X system. If you use another browser or operating system, you may see a slightly different representation from what is depicted here. In any case, the descriptions are still valid.

Organization

The individual chapters of this manual are assigned to four major Parts:

Part 1: Basics

This provides the bottom-line technical information needed to run Virthos. This is where we explain how to install and configure Virthos, how to obtain and activate a license, and how to save and restore data.

Part 2: Virthos Manager

Virthos Manager is the tool used for most administrative functions and all content editing—using easily manageable browser-based software. Virthos Manager is described in detail in Part 2. We will explain how to create and modify web pages, edit text, upload images, import and export content; and manage system settings, access rights, and users.

Part 3: Templates

This part of the manual is for web designers who want to design their own Virthos templates. This is where you will learn how to build and install templates, what template variants are and how to use them, and you will find an introduction to VirthosTalk, Virthos' template language.

Part 4: Integration

If you already have a website with pages that you will continue to use, you will learn how to make the old and new parts talk to each other. We will describe how to target specific Virthos pages directly, and the various ways in which a Virthos page can become the start page for a website.

PART 1

Basics

System Requirements

Virthos software has been designed to be usable on just about any of today's web servers. It is still a good idea, however, to verify that your server meets the requirements before installing Virthos. This chapter will tell you what you need to consider.

PHP Support

Virthos was developed using the PHP programming language. Therefore, the server needs to have a PHP module available, at the level of Version 4.1 or higher. PHP could be installed either as either a server or CGI module. Even the so-called “safe mode” that some providers activate on their web servers does not limit Virthos' functionality.

If you are not sure if your web server supports PHP, here is how you can find out:

1. **First, create a text file with the following contents:**

```
<?php
    phpinfo();
?>
```

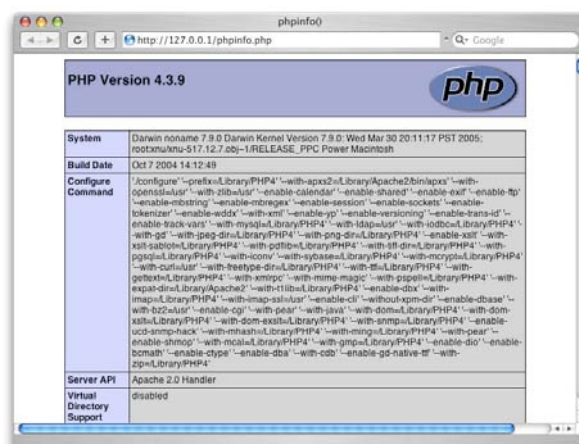
2. **Save this file as phpinfo.php.**

3. **Transfer the file to your web server's root directory.**

4. **Point your browser to the following address:**

<http://www.myserver.com/virthos/virthos.php>

Replace “www.myserver.com” with the URL of your web server. If your server supports PHP and you have followed these instructions carefully, your browser will display a detailed information page giving you many details about its PHP module.



If the title should mention a version number lower than 4.1, your version of PHP is too outdated to allow all of Virthos' functions to execute properly.

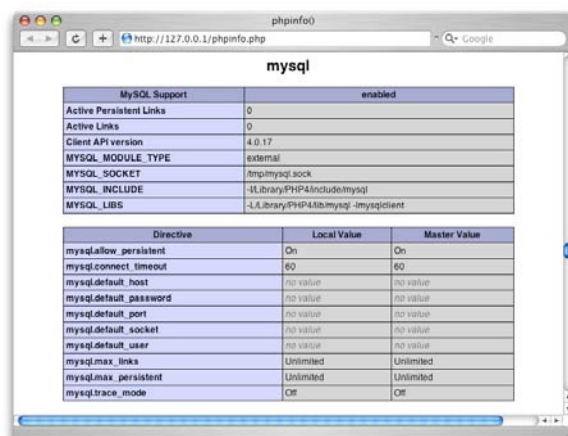
If you see an error message beginning with the words “parse error,” this does indeed confirm that your server has PHP support—but it also indicates that you have a typo in your “phpinfo.php” file. Please go back and check your spelling and try it again.

However, if you see the content of the phpinfo.php file as plain text, it is a sign that your server does not support PHP. It is also possible that the server requires all PHP scripts to be located in a specified subdirectory in order for them to be executed. If this happens to you, you should ask your web hosting provider for details.

mySQL Support

If you are installing Virthos' Basic version, you can skip this section. Otherwise, you will need to verify that any PHP scripts running on the web server can access a mySQL database server of Version 3.23.6 or higher, and that there is at least one mySQL database available for public use.

You can determine from the PHP information page whether mySQL has been installed at all. Scan down to the bottom of the page and see if there is a “mySQL” block that should look something like this:



MySQL Support		enabled
Active Persistent Links	0	
Active Links	0	
Client API version	4.0.17	
MYSQL_MODULE_TYPE	external	
MYSQL_SOCKET	/tmp/mysql.sock	
MYSQL_INCLUDE	-I/Library/PHP4/include/mysql	
MYSQL_LIBS	-L/Library/PHP4/lib/mysql -lmysqlclient	

Directive	Local Value	Master Value
mysql.allow_persistent	On	On
mysql.connect_timeout	60	60
mysql.default_host	no value	no value
mysql.default_password	no value	no value
mysql.default_port	no value	no value
mysql.default_socket	no value	no value
mysql.default_user	no value	no value
mysql.max_links	Unlimited	Unlimited
mysql.max_persistent	Unlimited	Unlimited
mysql.trace_mode	Off	Off

If you do not see it, then you will not be able to use Virthos Pro on this web server.

If you provide your own web server, you should in theory be able to create as many mySQL databases as you like. However, if you are using space provided on a remote web server, you may need to review the terms of your agreement with the provider to see if there is a mySQL database available to you. In particular, you will need to have the following information available:

- Address of the mySQL server (not applicable if the mySQL server is on the same computer as the web server)
- Name of the mySQL database that is available for you to use
- User name and password for accessing the mySQL database

If you cannot find this information in your web hosting agreement, you may have to request and pay for MySQL database access at an additional charge. Ask your provider about this.

Please note!

Virthos does not need to have exclusive access to the MySQL database. If you have already set up a database and have created scripts to access it (or are already using another version of Virthos to access it), you can still use the database. You may need to modify your Virthos configuration file to replace the default name prefix with another one (see the “Manual Installation” topic.)[¶](#)

Permissions

In order for Virthos to function properly, certain directories and files on the web server need to have their permissions set to specific values. This means that you must have a way to access the web server that will let you modify these permissions. In addition, PHP scripts must be able to create new directories and files on the server. Generally, these conditions must be met regardless of whether you provide the web server yourself or you use space made available by a provider. The best way to find out if the requirements have been met is to install Virthos and to see if it works.

Space Requirements

Thanks to its efficient design, the basic Virthos installation requires no more than 2 MB of disk space on the web server. You must add to this the space needed to accommodate your own HTML templates along with their associated resource files (images, CSS and JavaScript files). The content that is added will increase the amount of space needed over time. This applies especially to the images and documents that your authors upload to the server. On most websites, the space needed for text alone is not a major factor.

Installation

To use Virthos, you first need to create some directories on your web server and copy some files that are specific to Virthos. Permissions for these directories and files must also be set to specific values. All this can be done using the online installation program found at www.virthos.net, or manually.

Please note!

Virthos Pro will create some new tables in the MySQL database, and if tables with the same names already exist, their contents will most likely be deleted. In order to prevent something like this from happening, and to make it possible to run several copies of Virthos on the same machine, Virthos lets you specify various prefixes of your own choosing. Normally, each MySQL table that Virthos installs carries the prefix "vt2_". If your database already contains tables starting with this character string, you should specify a different string when configuring Virthos. If this applies to you, you should not use the Online Installation Program, but instead perform a manual installation, as described below. ¶

Using the Online Installation Program

The quickest way to install Virthos on a web server is to use the Online Installation Program. There are just a few requirements for using it.

- The web server on which you install Virthos must have a public IP address, either a numeric one, like "192.32.226.11," or a symbolic one, like "ftp.mycompany.com."
- The web server must be accessible by FTP. This means that it cannot be behind a firewall that blocks FTP access.
- You must be able to sign on to FTP with a password that allows you to write to the web server's hard drive.
- If you are going to install Virthos' Pro version, you must also know your MySQL access information (server address, database name, user name, and password).

If any of these requirements cannot be met, you should perform a manual installation of Virthos (see next section). Otherwise, you can use the Online Installation Program, available at

`http://www.virthos.net/services/installer`

The program will take you through the installation procedure step by step. At the end, a message will appear, stating whether your installation was successful or whether errors were encountered.

Since there are many different types of server, and it is impossible to test with all of them, it is possible that your online installation may have encountered some difficulties. In that case, you can perform a manual installation, as described in the next section. If your installation was successful, close the window, and continue as detailed under "Testing Your Installation."

Manual Installation

To install Virthos on a web server manually:

- 1. Copy the Virthos installation package to your hard drive.**

You may either copy the installation package from the Virthos CD or download it from the Downloads section of www.virthos.net. If you download the package, it will be a compressed Zip file, which you must first unzip. This will create a directory named `virthos` on your hard drive, containing the file `virthos.php` file and four subdirectories named `config`, `data`, `system`, and `templates`.

- 2. Next, customize the configuration file.**

Open the `config.php` file, which can be found in the `config` subdirectory of the `virthos` directory, with a text editor. This file contains a series of basic settings that Virthos requires. The meaning of the particular settings is thoroughly documented within the file. For security reasons, it is advisable to change the default master password to a new one. In addition, for the Pro version, you will need to enter the MySQL access data. The following chart will help you determine what to enter:

<code>mysqlServer</code>	Address of the MySQL server. In many cases, especially when a test environment has been set up on your own computer, it is sufficient to enter "localhost" or "127.0.0.1."
<code>mysqlDatabase</code>	Name of the MySQL database. You may obtain this name from your hosting provider or administrator when you request or create a new database. If you are working in a local test environment and have the authority to create a new database, you can enter any name here—Virthos will create the database automatically the first time that you invoke it.
<code>mysqlUser</code>	User name used by Virthos to connect to the MySQL server. You may also obtain this name from your provider or administrator. In a local test environment, you can enter "root."
<code>mysqlPassword</code>	Password for the user name. You will also obtain this from your provider or administrator. In a local test environment, you need to enter the root password that you used when installing the MySQL server.
<code>mysqlTablePrefix</code>	Prefix for the names of the tables that Virthos will be creating in the database. This prefix comes into play only if the database is also used by other Virthos installations or other PHP scripts. In such a case, you must enter a prefix that has not yet been used anywhere else.

3. Now, connect to your web server.

The procedure depends on where your web server is located.

- If you want to install Virthos in a test environment on your own PC, you only need to find the web server's root directory on your hard drive—that is, the directory in which the start page (usually *index.html* or *default.html*) is located.
- If the web server is located in a local area network, connect to the network location as you normally would. Please be sure that you have the correct permissions for creating directories on the web server, copying files, and granting access to others.
- If your web server is hosted remotely, use an FTP program to establish the connection. Here too, please make sure that you log in with a name that allows you to write directories and files.

4. Copy the Virthos directory to your web server.

Copy the virthos directory with its entire contents (see Step 2) to the root directory of the web server—the one in which the start page (usually *index.html* or *default.html*) is located. You can also copy Virthos into any subdirectory of the root directory.

5. Verify and correct the permissions of the directories and files that you have copied.

The files and directories that you copy to the web server automatically acquire certain permissions. What these are depends on how you have connected to the web server and what operating system it is running on. On Unix systems especially (including Linux and Mac OS X), if you have not set the permissions correctly, Virthos' functionality can be adversely affected. Therefore, please be sure that:

- The *data* directory within the *virthos* directory must provide write access for all users (`chmod 777`). The same is true for all files and directories within the data directory.
- All the other **directories** created in the installation process must be readable and executable for all users (`chmod 755`).
- All other **files** created during the installation must be readable by all users (`chmod 744`). In addition, the *virthos.php* file must be executable (`chmod 755`).

Some web servers may not actually require that the directories and .php files be executable (`chmod 755`), but in any case, it will not hurt.

This ends the manual installation process. To determine whether you have done everything correctly, please follow the instructions in the next section, under “Testing the Installation.”

Please note!

If PHP is installed on the web server as a CGI module (and not as an integrated server module), you must use a text editor to add a line at the beginning of the *virthos.php* file with the address of the PHP interpreter, for example: `#!/usr/local/bin/php`. If this provision applies to you, you must ask your provider or consult the documentation for your version of PHP to find out exactly how to formulate this line. In this situation, please pay close attention to the line breaks, and make sure that they are correctly coded in the *virthos.php* file. Most web servers prefer that their line breaks be coded for Unix files. Therefore, if you code this line in Windows or Mac OS, you should use a text editor that allows you to specify Unix line breaks. Another possibility is to use an FTP program that allows you to convert line breaks when uploading a file. ¶

Customizing the Installation

In order for Virthos to function correctly, some files and directories must have predetermined names, and they must be placed in specific locations. If you follow the instructions in this chapter precisely, you can be sure that you have met this requirement. There can, however, be cases where some variation on the default directory structure is required or desirable. These customizations are completely permissible, and do not adversely affect Virthos' functioning.

The following list details which changes to the directory and file structure are allowed, and which are not:

- The name of the Virthos directory—that in which the *virthos.php* file is located—can be freely selected.
- The contents of the Virthos directory can be transferred directly to the web server's root directory. This means that the *virthos.php* file, and the *config*, *data*, *system* and *templates* directories can be at the server's top level.
- The *virthos.php* file may not be renamed under any circumstances.
- The *data*, *data/database*, *data/files*, *data/media*, *system/resources* and *templates* directories may also be renamed or moved as described here. In this case, you must modify the configuration file to indicate the correct path (see the following section).

There are special rules that apply to the directory structure within the templates directory. This is described in the “Creating and Installing Templates” chapter in Part 3.

Please note!

If you rename or move files and directories, this can have an effect on the address with which the *virthos.php* file is associated. This could cause a license key that has already been issued to lose its validity. Please make any planned modifications before you order a license key. ¶

Special Cases

Some web servers have special requirements for PHP scripts. For example, for security reasons, some servers are configured so that PHP scripts can only be executed if located in a specific directory (often called *cgi-bin*), while other files such as images and HTML pages may not be located in this directory. This will not stop Virthos from functioning, but you will need to modify the directory structure and document this change in the *config.php* file.

Since all particular situations can neither be anticipated nor described, we will just concentrate here on one example: how to install Virthos on a web server where all PHP scripts need to be located in the */cgi-bin* directory, although images, style sheets, etc. may not be located in this directory.

1. Install Virthos in your web server's root directory.

Follow all the instructions that were previously outlined.

2. Copy the entire Virthos installation directory to the cgi-bin directory.

Make sure that you are copying the folder, and not just moving it. The result should be that there are two identical Virthos directories on the server.

3. Delete the superfluous files and directories.

The following files and directories can be deleted from the Virthos directory in the *cgi-bin* directory: *data/files*, *data/media*, *system/resources*

The following files and directories can be deleted from the Virthos directory in the web server's root directory: *config*, *data/database*, *templates*

4. Next, customize the configuration file.

Load the *cgi-bin/virthos/config/config.php* file on your computer, open it in a text editor, and add the following lines or modify them if they already exist:

```
pathToFiles = ../../virthos/files/
pathToMedia = ../../virthos/media/
pathToTemplates = ../../virthos/templates/

baseURLFiles = http://www.mycompany.com/virthos/data/files/
baseURLMedia = http://www.mycompany.com/virthos/data/media/
baseURLResources = http://www.mycompany.com/virthos/system/resources/
baseURLTemplates = http://www.mycompany.com/virthos/templates/
```

On the lines beginning with *pathTo*, replace the three dots with the path to the web server's root directory. You can use the *phpinfo.php* file, described under "System Requirements" above to determine this path. You will find the path in the line beginning with

```
$_SERVER[ "DOCUMENT_ROOT" ]
```

The line is located in the last block on the page.

On the lines beginning with `baseURL`, replace `www.mycompany.com` with the URL of your web server.

5. Save the configuration file and transfer it to the web server.

Please be sure to copy the file to the correct Virthos directory, namely the one in `cgi-bin/virthos/config/`.

This ends the special installation process. You will find out if you have performed all these steps correctly by following the instructions in the following topic, "Concluding and Testing the Installation."

Concluding and Testing the Installation

To conclude the installation and determine if you have successfully installed Virthos:

1. Execute the Virthos.php script in your browser.

Enter the following address:

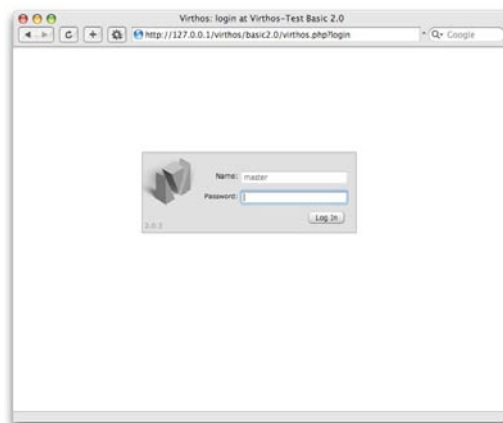
`http://www.myserver.com/virthos/virthos.php`

Replace `www.myserver.com` with the address of your web server, or if testing on your own computer, use `127.0.0.1` or `localhost`). If you did not install Virthos in your web server's root directory, after the server address, include the path to the installation directory.

Please note!

If the web server is located on your own computer, you cannot simply open the file `virthos.php` in your browser, which would be the equivalent of an address beginning with `file://`. You must use the HTTP protocol to access the file—this means that the URL must begin with `http://`.

If your installation was successful, after entering the correct address in your browser, you will see Virthos' Login window:



If you should see one or more error messages, please verify that:

- you have entered the correct address
- you have made sure that all the permissions are correct
- your web server meets all the requirements
- If you have installed the Pro Version: Is the MySQL access information in the *config.php* configuration file correct?

2. Enter the master password.

The first time Virthos runs, it must perform some basic operations— for example, in the Pro version, it must set up the MySQL database). For security reasons, only the master user can do this. Enter the master password that is stored in the *config.php* configuration file. If you used the Online Installation Program, the master password is the same as the FTP password that you used during the installation process.

3. Click the [Log In] button.

You will see the Virthos License Agreement. If, when using the Pro version, you see an error message instead of the License Agreement, please verify that:

- there are not already database tables belonging to another Virthos installation with the same prefix in the database.
- the MySQL user you have specified has the right to create new database tables.

If necessary, correct any entries you have made in the configuration file, and make sure that MySQL server complies with all requirements. Then start over with Step 1.

4. Confirm your acceptance of the License Agreement.

Please read this agreement carefully. If you cannot agree to the terms, you cannot use Virthos. Otherwise, check the box at the bottom that signifies your acceptance of the terms of the Agreement, and click [Continue].

After you accept the Agreement, you will see Virthos Manager appear in your browser window. Close this window for now. Continue as detailed in the chapter “Getting Started” in the “Introduction to Virthos.”

Configuration

Virthos' behavior is controlled by a set of so-called configuration parameters that can be found in the configuration file, *config.php*. This chapter describes how to use the configuration file and the individual parameters within it.

Please note!

Under normal circumstances, you will not need to modify the configuration file. You should do so only if specifically instructed by this manual. Mistakes in settings in *config.php* can seriously degrade Virthos' operation. Therefore, you should only make changes if you know exactly what you are doing. In that case, you should still make a backup copy of the file first, so that you can revert to the old (working) settings in an emergency. ¶

General Information About the Configuration File

The *config.php* file is located in Virthos' *config* directory. Even though its name ends in *.php*, it is not a PHP file in the true sense. The file extension serves to safeguard its contents— it ensures that the file cannot be downloaded from the server and thus keeps the master password secure.

Each line of the configuration file can contain only one configuration parameter. The parameters are specified according to the pattern:

parameterName = *parameterValue*

On the left side of the equals sign is the name of the configuration parameter, and on the right is the value to be assigned to the parameter. The spaces on either side of the equals sign are optional. You may optionally enclose the parameter value in double quotes:

parameterName = "*parameterValue*"

Virthos ignores any lines that do not contain an equals sign. It also ignores lines beginning with a semicolon. This makes it easy to include comments in the configuration file or to “comment out” individual lines. Instead of deleting an assignment, you can put a semicolon at the beginning of the line to deactivate it. You may then easily re-enable the statement by removing the semicolon.

Predefined Configuration Parameters

There are a number of parameters that have a special meaning to Virthos. These are described both in the configuration file itself and in various places in this manual. Here are a few of them:

<i>masterPassword</i>	The password that you use to log in to Virthos as the master user (with user name "master.")
<i>mysqlServer</i>	The address of the MySQL server that you want Virthos to use (Pro version only).
<i>mysqlDatabase</i>	The name of the database that Virthos is to use (Pro version only).
<i>mysqlUser</i>	The user name that Virthos is to use to log in to the database server. (Pro version only)
<i>mysqlPassword</i>	The password that Virthos is to use to log in to the database server. (Pro version only)
<i>mysqlTablePrefix</i>	String of characters placed at the beginning of all MySQL tables (Pro version only).

User Configuration Parameters

You can also use the configuration file to define global variables that you assign yourself with the VirthosTalk placeholder `vtGlobal` (see the VirthosTalk Reference Manual). You can use any parameter names that you wish, as long as they do not contain accented, punctuation or special characters (except for the hyphen and underscore characters).

If you use your own settings pages (see the chapter "Managing System Settings") and you define the same variable both on your settings page and the configuration file, the assignment using the configuration file always takes precedence. In other words, any time you assign a value to a variable in the configuration file, assignments using a settings page remain inoperative.

Licensing

Virthos licensing is not based on the computer on which the Virthos software is installed, but on the address for which the *virthos.php* script is invoked. There are two distinct situations: Are we dealing with an address on the same computer (local access) or an Internet or intranet address (remote access)? The following sections describe the two situations in detail.

Local Access

Virthos considers local access to be in effect when the *virthos.php* file is invoked for the address *127.0.0.1* or the host name *localhost*. In other contexts, the word “local” could refer to other situations besides these two, but in respect to Virthos licensing, only addresses mentioned above are considered to be “local.”

When accessed locally, Virthos always acts like the registered version—that is, there are no advertising displays or limitations of functionality. Your ability to use the software is not limited in its duration, either. You may use Virthos as often and as long as you wish.

Remote Access

Remote access, to Virthos, is considered to be in effect when the *virthos.php* file is not invoked using the address *127.0.0.1* or the name *localhost*. This applies also to access over a local network—even if the browser is located on the same machine as the Virthos software.

Under remote access, Virthos runs initially in demo mode. The demo version contains the same functionality as the registered version, except that at the beginning of each session, a window appears stating that Virthos is running in demo mode. There are no other differences in functionality or time limitations.

The demo version can be converted to the registered version by entering a license key on the System Settings page. (See the “License Settings” topic in the chapter “Managing System Settings.”) This key always refers to a specific address: Only when Virthos is invoked with that address will it operate in registered mode. If it is invoked with any other address, the demo notice will appear.

For example:

If you have both a *.com* and a *.net* domain assigned to you, and you order a license for

`www.mycompany.com/virthos/virthos.php`

and you refer to

`www.mycompany.net/virthos/virthos.php`

Virthos will show the demo notice. You could avoid this by acquiring another license for your .net domain, but that would entail payment of another license fee. The other, more economical, option would be to redirect visitors to the .net domain's start page automatically (or with a single click) to the .com domain.

Please note!

There is one exception to this rule: If you purchase a license key for address beginning with *www.* (such as *www.mycompany.com*), Virthos will still operate in registered mode if someone leaves out the *www.* at the beginning (in this example, *mycompany.com*). However, your provider may not allow your site to be accessed without the initial *www.* ¶

Ordering and Registering License Keys

To order and enter a license key:

1. Go to “Licensing” on the System Settings page.

To reach this page, refer to “Managing System Settings” in Part 2 of this manual. Please be sure to invoke Virthos using a non-local address, and to log in as the master user. Otherwise, you will not be able to carry out the following steps:

2. Click [Order License Key].

You will be directed to an order form on the Virthos website. The Virthos version number and the current address will automatically be entered in the form.

3. Fill out the order form and follow the additional instructions given there.

When your order is complete and you have submitted your payment, you will receive a confirmation that will include the license key.

4. Go to “Licensing” on the System Settings page.

Be sure to invoke Virthos using the same address as in Step 1.

5. Click [Register License Key].

A dialog window where you can enter the license key appears.

6. Enter the license key and click [OK].

Be careful to enter the license key exactly as it appears in the confirmation.

That concludes the licensing process. From this point on, Virthos will always appear in registered mode for the licensed address. If you need license keys for additional addresses, just repeat these steps for each one.

Maintenance

Whether you run your own web server or use space provided by a web hosting service, you need to back up the data that your website uses at regular intervals. If you should lose data on the server because of a technical malfunction, you will then be able to quickly replace the lost data.

Backing up Data

To back up all the files that are essential for Virthos' operation:

- 1. Create a backup of the database contents (Virthos Pro only).**

In Virthos' Pro version, all text content, as well as user and system settings are stored in a MySQL database. As a rule, you will not be able to access the database tables directly; therefore, Virthos makes available a process that backs up all these tables in the Virthos directory. You can enable this process by entering the following address in your browser (replace the three dots by the name of your web server, and add any applicable path information):

`http://.../virthos.php?backup`

When the backup is finished, the Virthos start page will be displayed. The backup file will be stored on the web server in its *data/database* directory and will carry the name *backup.php*. (This backup copy is not, as one might expect, a PHP file. (The *.php* extension is used only for reasons of security.)

- 2. Copy the Virthos directory to another drive.**

Virthos is designed so that all the files that it needs to use are normally located in a single directory. This directory, assuming you have not customized your installation, is called "virthos" and contains the *virthos.php* script file as well as the four subdirectories *config*, *data*, *system* and *templates*. Copy this directory and all of its subdirectories to another computer or another hard drive. If the web server is located on your local network, a simple copy will do. Otherwise, use an FTP program to transfer the data.

Please note!

In addition to the method described in Step 1, Virthos Pro has another way of backing up the contents of the MySQL database. This is described in Part 2 of this manual in the chapter entitled "Managing System Settings." ¶

Restoring Data

If you should ever lose the data on your web server, you can restore it in the following manner:

1. **Copy the last backup file back to the web server.**

Reverse the process that was described in the above section, and copy the Virthos directory created by the backup, with all of its subdirectories and files, to the correct location on the web server. If the web server is located on your local network, a simple copy will do. Otherwise, use an FTP program to transfer the data.

2. **Next, adjust the permissions if necessary.**

Depending on how you transfer the data from and to the web server, it might be necessary to adjust the permissions connected with some directories and files. In that situation, the following rules apply, just as when installing Virthos (see the “Installation” chapter.)

3. **Restore the database contents if necessary (Virthos Pro only).**

In case your database contents are ever lost, the Pro version provides a process for restoring the database. To start this process, invoke Virthos with the `restore` command, like this:

```
http://.../virthos.php?restore
```

For security reasons, only the master user is allowed to restore the database. You will therefore be asked for the master password. The restoration process, which can take several minutes for large amounts of data, is then started. At the end, you will see the Virthos License Agreement. Confirm your acceptance again, and you will soon be able to work with Virthos as before.

Backup Strategies

How often you save any particular data will depend on how often the content on your web server changes. Some common practices that have evolved include these:

Virthos Basic

Virthos' Basic version does not use a MySQL database. Therefore, all content is saved directly and exclusively to the Virthos directory. Each time you complete a major addition or revision of your website, it is advisable to copy the Virthos directory from the web server to another computer. If you have not made any changes to templates or configuration files, it is sufficient to copy only the data directory.

Virthos Pro

In Virthos' Pro version, all changes to content are first saved in the MySQL database. It is advisable at the very least to save the database contents on the web server. Since simply invoking a URL is all it takes to start the backup process, it is easy to automate it. In Unix systems, for examples, you can create an entry in the operating system's *cron* table to perform this task.

Releasing Unused Disk Space

Virthos normally releases any disk space that has been allocated on the web server or the MySQL server as soon as it is no longer needed. In unusual cases, there might be some bits of data left on the server's hard disk that are no longer needed. To instruct Virthos to delete this data and make the space available again, invoke the Virthos script with the following parameter:

```
http://.../virthos.php?-act=tidy
```

Virthos will then carry out its cleanup operation and again show the start page. It will not explicitly say that the operation has completed.

Please note!

In Virthos' Pro version, this cleanup operation will automatically take place when you start a data backup. If you regularly back up data, you do not need to be concerned about freeing up disk space. ¶

Restoring Default Settings

At any time, you can reset Virthos to its state at the end of the installation process. The procedure depends on which version of Virthos you are running.

Virthos Basic	Empty out the <i>database</i> subdirectory, which you will find in the <i>data</i> directory.
Virthos Pro	Drop the tables of the MySQL database that begin with preset prefix. The easiest way is to use the “dbdestroy” command, described in the “Uninstalling” chapter.

The next time you invoke the Virthos script, the original database contents will be restored, and you will be asked to accept the License Agreement once more.

Domain and Server Replacement

Virthos' backup and restore processes can be used to move a Virthos installation from one web server to another. Such a transfer might be necessary if you change your hosting provider, or if you build a website in a test environment and later want to move it to the actual web server.

To move a complete Virthos installation from Server A to Server B:

1. **Create a backup on Server A (as described in “Backing up Data”).**
2. **Customize the saved configuration file (Virthos Pro only).**

Probably Server B will have different access data for its mySQL database from Server A's. Make the needed adjustments to the *config.php* file. The configuration file is located in the backed up Virthos directory's *config* directory.

3. Restore the data to Server B (see the “Restoring Data” topic).

Remember to adjust the permissions for the *data* directory and the files and subdirectories contained in it.

Uninstalling

To permanently delete Virthos from a web server:

- 1. Create a backup copy of your Virthos content.**

If you are absolutely sure that you will never again need any content created with Virthos, you can skip this step. Otherwise, follow the steps outlined in “Backing up and Restoring Data” in Part 3 of this manual.

- 2. Drop any mySQL database tables that you no longer need.**

The following applies only to uninstalling Virthos Pro: Invoke Virthos with the “dbdestroy” command.

```
http://.../virthos.php?dbdestroy
```

In the window that appears, enter the master password and click [Log In]. The tables used by Virthos are then irrevocably dropped from the mySQL database. All other database content will remain.

- 3. Delete the “virthos” directory created at installation time, along with all files and subdirectories contained within it.**

If you use FTP to access the web server, be sure to log in with the same user name as when you installed Virthos to ensure that you have the necessary permissions for deleting data.

PART 2

Virthos Manager

Introduction

Virthos Manager is Virthos' principal work environment. It is used for both content editing and administrative tasks. It can be used by any computer that

- has a web browser
- can connect to the Internet or to the web server
- has Virthos installed on it.

To use Virthos Manager, you must have the Virthos master password, or (for Virthos Pro) a combination of user name and password that allows access to Virthos Manager.

Please note!

Since Virthos Manager makes significant use of JavaScript functions, you should use a recent-edition web browser to invoke Virthos Manager. Virthos Manager has been successfully tested with the following browsers: Internet Explorer 5.5 and 6.0 (Windows), Firefox 1.0 and 1.5 (Windows and Mac OS X), and Safari 1.3 and 2.0 (Mac OS X), as well as others. If you use a different web browser or an older version, Virthos Manager may not function exactly as expected. ¶

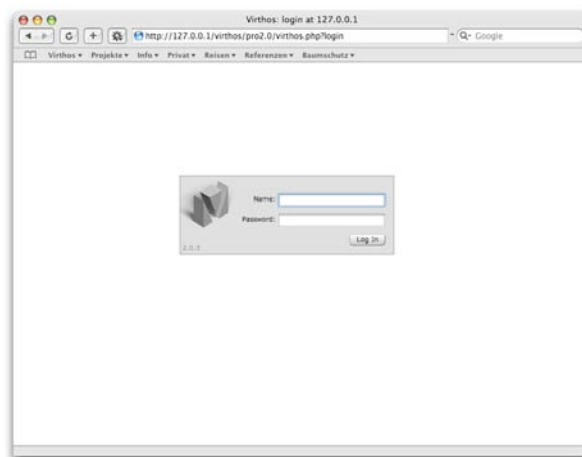
Starting and Stopping Virthos Manager

Virthos Manager is started automatically when you log in to Virthos. The login page can be accessed through the following URL:

`http://.../virthos.php?login`

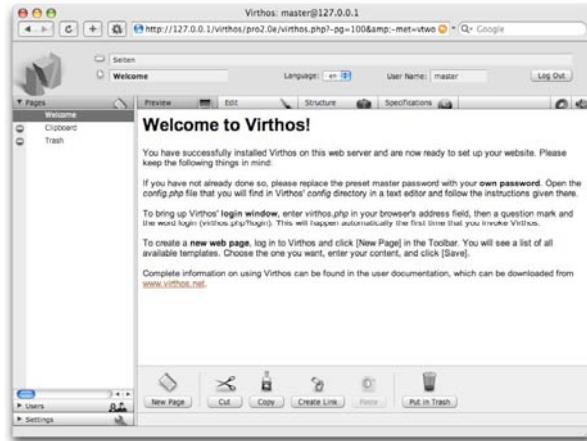
You must replace the three dots by the address of the web server and the path to the *virthos.php* script.

If you entered the address correctly, you will then see the login window.



In the Basic version, since there is no user administration, the user name field is automatically populated with the word “master.”

Enter the correct user name and password and click [Log In]. If the name and password are correct, Virthos Manager will open in your current browser window. This may take several seconds, depending the speed of your web server and your Internet connection. You will know that it has been loaded once you no longer see the “turning wheels.”



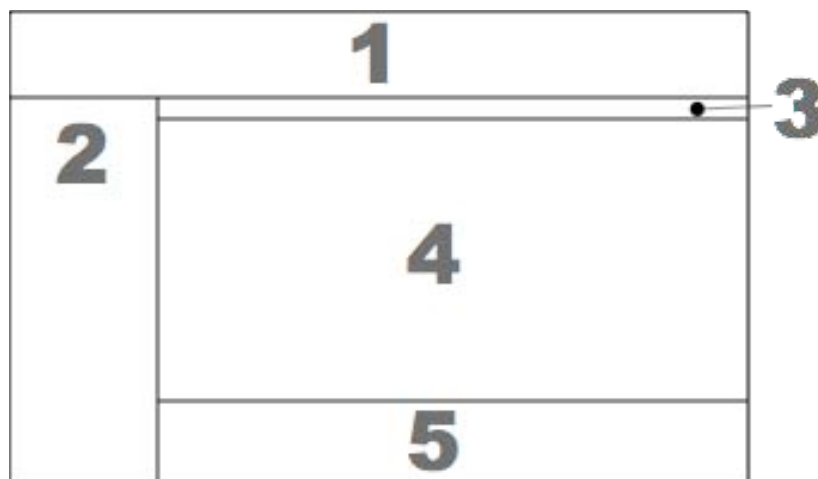
To exit from Virthos Manager, click [Log Out] in the upper right corner, and confirm your intention in the window that appears afterward.

Please note!

You should always explicitly terminate Virthos Manager, and not just close the browser window. Otherwise you may open your Virthos system up to unauthorized use. ¶

Virthos Manager's Regions

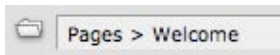
The following illustration depicts Virthos Manager's main regions schematically:



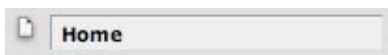
1 = Header Area, 2 = Navigation Bar, 3 = Mode Selector, 4 = Content Area, 5 = Toolbar

The Content Area always shows the latest page selected, which is displayed in the way specified by the Mode Selector. The Content Area also displays other informational and system pages, as well as forms that are not associated with any specific mode, when necessary. Here we will cover each of Virthos Manager's other areas in detail:

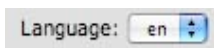
The Header Area



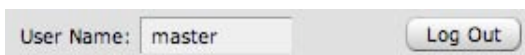
At the very top of the Header Area, the path to the current page is shown. The path is a list of all the pages leading to the current page in the hierarchy. The path information serves not only to show the context, but also as a navigation aid. By clicking any of the elements in the path, you can make that the current page. Clicking on the folder symbol to the left of the path display makes the immediate parent of the current page (shown at the end of the path) replace the current page.



Below the path is the name of the current page. Clicking on the file symbol to the left of the name causes the current page to appear in a new browser window. This will immediately give you an idea of how the page would appear to a guest user.



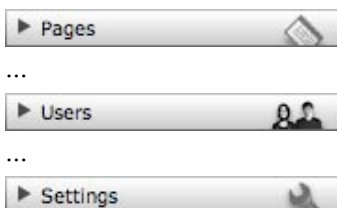
To the right of the page name (in the Pro version) is a list of available language environments. You can use this to quickly move from one language to another.



At the far right of the Header Area you can see the user name that you used to log in (in the Basic version, this will always be “master”). Also, by clicking on [Log Out], you can quit Virthos Manager.

The Navigation Bar

The Navigation Bar contains a list of all the pages available through Virthos Manager—not only the pages that make up your website, but also the “system pages” that are not part of your site, but are used by Virthos for its internal purposes. To give a clear overview of the available pages, they are grouped into categories that can be explored by clicking one of the following buttons:




The list of available categories is determined by your Virthos version—for example, Virthos Basic does not display a “Users” category.


When you start Virthos Manager, you initially see only the “Pages” category. The buttons for the other categories can be found at the bottom of the Navigation Bar. By clicking one of the other categories, you will open that one, and the previously opened category is closed. In Virthos Pro, you must be logged in as a user with Administrator rights in order to open the “Users” or “Settings” category.

Within the open category, the currently selected page is shown in a darker color. Clicking on any entry in the list will display that particular page. The same is true for the category lists themselves: For example, you can click the “Pages” button, and the button will turn black, and when you click the button again, the contents of the relevant page (in this case, the built-in Virthos start page) will be displayed in the Content Area.

Showing and Hiding Subpages

A gray triangle to the left of the page name () indicates that the page contains subordinate pages, also referred to as subpages. Clicking on this triangle once will cause the subpages to appear; clicking again will eliminate them from the display. When there is a large number of subpages, the list is not loaded into the Navigation Bar in order to cut down on the load time. In this case, you will see a white triangle with a gray border, and clicking on it will not have any effect.

Locked Pages

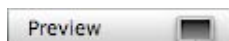
If you see a “no entry” symbol () next to a page, this means that the page can be opened only by users who are logged in, and that this page is not accessible to the general public. This applies to all system pages, but also to content pages whose access rights have been modified (see the chapter entitled “Managing Access Rights”)

Reloading the Navigation Bar

Whenever you create new pages, move existing pages, or change page specifications, Virthos reloads the content of the Navigation Bar in order to show their current state. However, there may be times where this automatic process is not enough, as when another user has changed some content in the meantime. If this happens, you can force reloading of the Navigation Bar by clicking the “Pages” symbol at the right end of the “Pages” button.

The Mode Selector

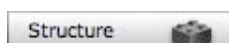
The Mode Selector contains several buttons that let you select the mode in which the current page is displayed. The current mode is shown as a darker button.



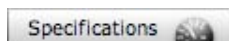
Preview Mode displays the page more or less in the same way that it will appear on the Web. Depending on the template on which the page is based, some aspects of the page are visible only in Virthos Manager. By clicking the File symbol (to the left of the page name in the Header Area), you can see a "true" preview.



Edit mode makes the content of the current page available for editing. Text can be edited, and images can be uploaded or replaced. The template on which the current page is based determines exactly what options are available.



In Structure mode, all subpages of the current page—meaning all pages subordinate to it in the hierarchy—are listed. Next to the page name is the name of the template on which it is based, as well as the creation and last modification dates. In Structure mode, the order in which the subpages are listed can be modified by clicking one of the column headings, such as "Pos," "Name," "Created," or "Modified" (see "Changing Sort Criteria" in the chapter "Managing Pages"). You can also mark individual pages in order to perform various operations on them (see "The Toolbar").



The Specifications mode displays general information about the current page, and gives you the opportunity to modify the page name, the sort order or access rights (see the chapter entitled "Modifying Page Specifications").



Clicking on this "gear" symbol will cause a list of special functions applicable to the current page to be displayed. Some of these functions are available for all pages, and others are only available for pages based on a particular template.



Clicking on this "Virthos logo" symbol will result in the display of information about the Virthos version and the server environment.

The Toolbar

Depending on the mode in effect, the Toolbar displays various tools that you can use to carry out different actions. Sometimes you will also see error messages and other responses from Virthos below the Toolbar.



Clicking on the "New Page" tool produces a list of available templates and lets you create a new page based on the selected template (see "Creating New Pages" in the chapter entitled "Managing Pages").



Behind these tools are some clipboard functions that you can use to move pages, or create copies or links. The first three tools in Preview mode refer to the current page. In Structure mode, they refer to the pages currently highlighted (refer to "Moving Pages," "Copying Pages," and "Creating Links" in the chapter "Managing Pages").



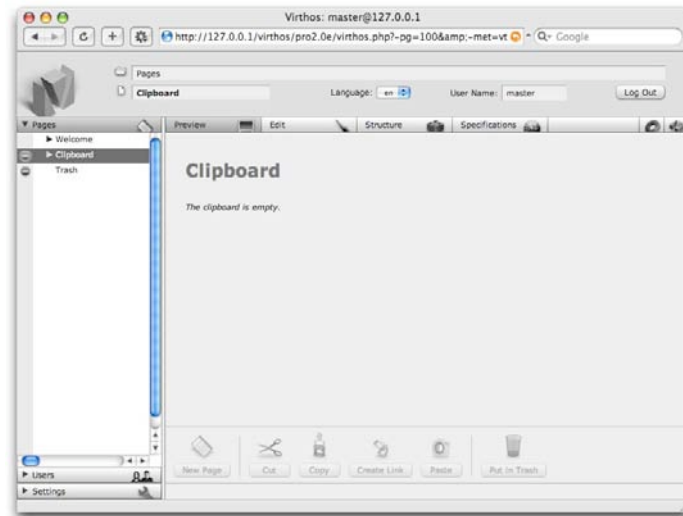
In Preview mode, this tool can be used to put the current page in the Trash Can. In Structure mode, the result is that all highlighted pages will be placed in the trash (see "Removing Pages" in the chapter "Managing Pages").



These tools become available when a form is present in the Content Area. You can use them to save data entered in forms or revert to the previous view without saving the data.

The Clipboard

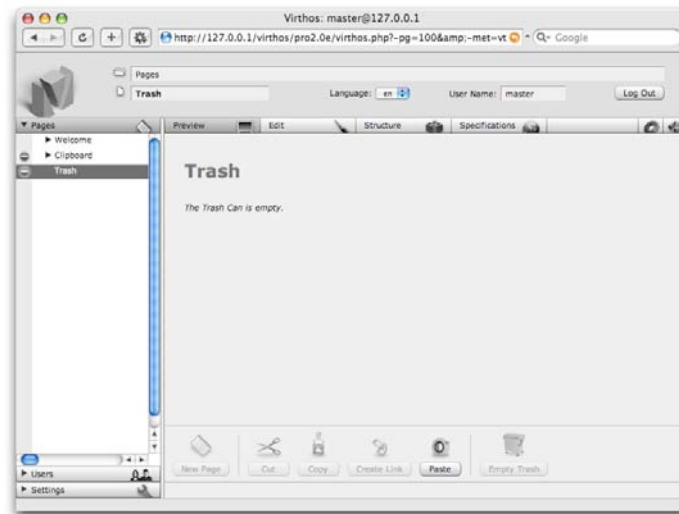
The Clipboard is a facility that Virthos provides for moving or copying pages or creating links (see the corresponding sections in the chapter entitled “Managing Pages.” In the Navigation Bar, the Clipboard is shown just like any other page; by clicking on its name, you can make it the current page.



To see which pages are currently on the Clipboard, you can list its contents using Structure mode. Click [Structure] in the Mode Selector. Structure mode also lets you remove individual pages from the Clipboard.

The Trash Can

Just like the Clipboard, the Trash Can helps you manage editing operations (see “Removing Pages” in the chapter “Managing Pages”). The Trash Can appears just like a normal page in the Navigation Bar. By clicking on the words “Trash Can,” you can make it the current page.



It is a good idea to empty the Trash Can from time to time, so that the disk space used by the pages in it can be freed. All you need to do is click [Empty Trash] on the Toolbar. To see which pages are currently in the Trash Can, you can list its contents using Structure mode. Click [Structure] in the Mode Selector. Structure mode lets you retrieve individual pages from the Clipboard (by clicking [Cut]) or delete them completely (by clicking [Delete]).

Managing Pages

Virthos Manager is essentially an editing tool. It makes it possible to edit the contents of pages on the Web, create new pages, and restructure a website. This chapter gives a detailed description of the possibilities and how to make use of them.

Creating New Pages

Creating new pages in Virthos is a two-part process: First, you select the template that the new page will be based on; then you fill in content in the parts of the template marked as editable. The exact procedure is:

1. Navigate to the page that will be above the new page in the hierarchy.

The new page will then be subordinate to the page shown by Virthos Manager. Of course, you can use the clipboard later to move the new page to a new location, but it is easier to create it in the correct place to begin with.

2. On the Toolbar, click [New Page].

You will see a list of the available templates: The size of this list and which templates are in it depends on two factors:

- which templates have been installed and
- which template the current page is based on.

If the current page's template has predetermined that all subpages should be based on a specified template, this step is skipped, and you will immediately see a content entry form (see Step 4).

At first, Virthos lists only the templates in the same package as the current page's template. In order to see the templates belonging to other packages, click [Show all Packages]

3. Click the name of a template to select it.

A form with entry fields for all areas that are marked editable in the template will appear. If there are no editable areas in the template, the form will ask you only to specify the name of the new page.

4. Fill out the form.

Insert your desired content. You can enter text directly into the text fields. For images and other external data, you can choose files on your hard drive that will be uploaded to your server when you submit the form. More details are available under "Editing Text" and "Uploading Images and Files."

5. On the Toolbar, click [Save].

This action submits the form, and all the content you have entered is transmitted to the server. You will then see the page you have just created in Virthos Manager's Content Area.

A newly created page at first takes on the access rights of its parent page. If the parent page was accessible for viewing by the public, the newly created page automatically is, too. If you do not want it to be, you need to alter the access rights accordingly after creating the page (see “Modifying Page Specifications”).

Please note!

When you create a new page, it is first automatically given a name based on the text appearing on it. To assign a different name to it, select Preview mode after you create the page (see “Modifying Page Specifications”). ¶

Changing Page Content

At any time, you can modify the content you placed on a page when you created it. Just proceed as follows:

1. Navigate to the page whose content you would like to modify.

You can only change the content of the current page—meaning the page you are viewing in Virthos Manager and whose name is visible in the Header Area.

2. In the Mode Selector, click [Edit].

Virthos Manager's content area displays a form that you can use to edit the contents of the current page. Depending on the template on which the current page is based, you can edit the text or upload images or documents. More details are available under “Formatting Text” and “Uploading Images and Files.”

3. Click [Save] on the button bar.

This will submit the form and transmit all modified content to the server. You will then see the page you have just created in the Virthos Manager's Content Area.

When you change the contents of a page, these changes take effect immediately. This means that they are visible to visitors as soon as you save the new content. If you do not want this to happen, then you should first copy the page concerned to a location that is not publicly accessible and make your changes there (see “Copying Pages”). When you are ready to make your changes public, you can delete the old version of the page and move the copy to its place.

When you are planning a comprehensive update involving a great number of pages, copying individual pages can be an inefficient approach. In this case, you can use Virthos' export and import functions to make a copy of the entire website or portions of it (see “Exporting Pages” and “Importing Pages”).

Please note!

If the page being edited has a name that was automatically assigned, after you edit the page contents, the name could change. This can be an unwanted result, but it can also be prevented if you set the “Protect Page Against Moving and Renaming” option in Page specifications on (see “Modifying Page Specifications”). As soon as assign a name to a page manually, its name will no longer change automatically. ¶

Translating Page Content

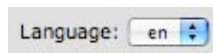
Virthos Pro gives you the ability to render each page in various languages. The visitor is thus able to select a language and see the content in the correct language version—as long as that version is available. Before you can enter content in various languages, you must first set up each in language in System Settings (see the chapter entitled “System Settings”).

To translate a page:

1. **Use Virthos Manager to navigate to the page to be translated.**

2. **Select the desired language.**

In Virthos Manager's Header Area, you will find the language selection menu to the right of the page name. This menu lists all languages that were previously set up in System Settings:



After you select a language, the current page is displayed in the chosen language—as long as there is an existing translation. Otherwise, the content remains unchanged.

3. **In the Mode Selector, click [Edit].**

This will bring up an editing form for the current page. If no translation is yet available, the contents will be displayed in the default language.

4. **Replace the contents with the desired translation.**

You can overwrite any portion of the text and replace images or other files or remove them from the page entirely.

5. **On the Toolbar, click [Save].**

This will submit the form and store the translation in the Virthos database. You will then see the current page in its translated version. When you click on the original language on the Language menu, you will see that the original contents are still there.

To translate several pages in succession, you only need to select the new language once. Virthos will keep the setting active until you again change it explicitly.

Moving Pages

Virthos maintains each page's position in a tree structure. This means that each page is subordinate to another, and may have its own subordinate pages underneath it. The page above a given page in the hierarchy is referred to as its “container page,” and the subordinate pages are known as “subpages.”

You can change the Virthos tree structure by assigning a different container page to one or more pages:

1. **Bring up the current container page of the pages that you want to move.**

If you want to move only a single page, bring up this page, and skip Steps 2 and 3.

2. In the Mode Selector, click [Structure].

The content area now lists the subpages of the current page.

3. Check to pages to be moved.

For each page to be moved, check the box at the far left. If you want to move all the pages, check the box at the top of the list, and this will cause all the pages to be checked.

4. On the Toolbar, click [Cut].

The current page or the checked pages are moved to the Clipboard. They will temporarily leave their former position.

5. Bring up the new container page.**6. On the Toolbar, click [Paste].**

The pages on the Clipboard are moved to the new location.

If the pages that you want to move are currently located in different places, you can cut each of them separately, one after the other, and then place them together in the new location.

Copying Pages

Instead of building a new page from the ground up, you can make a copy of an existing page and then modify it in the appropriate places. (If you need an exact copy that automatically picks up any changes that you might later make to the original page, please read the topic “Creating Links”).

To make a copy:

1. Bring up the page that you want to copy.

If you want to copy several pages at the same time, bring up the container page and in Structure mode, check the pages to be copied (see the topic “Moving Pages”).

2. On the Toolbar, click [Copy].

A copy of the current page is placed on the Clipboard.

3. Bring up the page that will be the container page for the page you are copying.**4. On the Toolbar, click [Paste].**

The copy on the Clipboard then becomes a subpage of the current page.

Please be aware that copying involves only the contents of a page—not the contents of its subpages. This can cause a page that contains a list to appear different from the original. If you want to copy a page and its subpages, instead of copying them as described here, it is better to use Virthos' export and import functions instead (see the topics “Exporting Pages” and “Importing Pages”).

Creating Links

If the same page needs to appear in various places in the Virthos tree structure, it is better not to create multiple copies of it. If you needed to make changes to the page in the future, then each copy would have to be edited separately. This would not only be inconvenient, but would carry with it the risk that you might overlook some of the copies and leave some of them with obsolete contents.

With Virthos, there is an easier way—creating links. A link is like a copy. The difference is that a link automatically replicates the original page and thus guarantees that the page always appears up to date. From the technical point of view, a linked page has no content of its own; it contains only a reference to the original page.

To create a link for a given page:

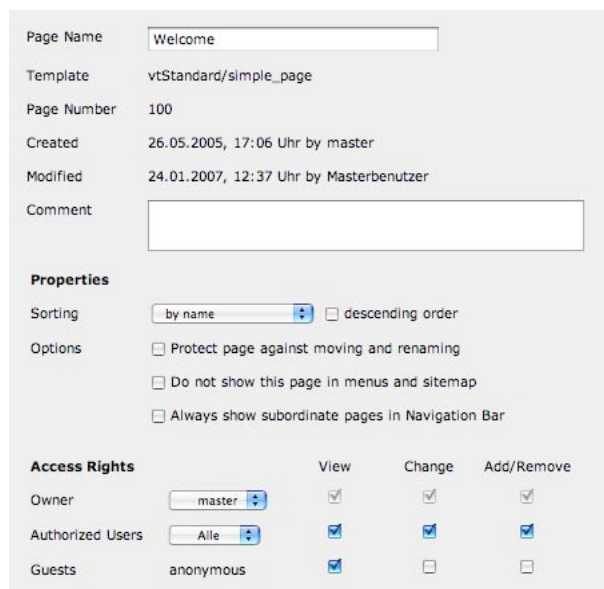
- 1. Navigate to the page for which you want to create a link.**
If you want to create links for several pages at the same time, bring up their container page and in Structure mode, check the pages for which you want to create links (see the topic “Moving Pages”).
- 2. On the Toolbar, click [Create Link].**
A link to the current page is placed on the Clipboard.
- 3. Bring up the page that will be the container page for the linked page.**
- 4. On the Toolbar, click [Paste].**
The link on the Clipboard then becomes a subpage of the current page.

To make it more apparent that a page is linked, Virthos Manager shows it in italic type in the Navigation Bar. You can move, copy, rename or delete a linked page. The original page is not affected. Please note, however: If you edit the contents of a linked page, you will also be modifying the contents of the original page!

Modifying Page Specifications

In Virthos, each page has certain properties and specifications that you can change as you see fit. For example, you can rename a page, modify its access rights or establish a certain sort order for its subpages. To change these and other specifications:

- 1. Bring up the page whose specifications you would like to modify.**
- 2. In the Mode Selector, click [Specifications].**
This will bring up the properties of the current page in Virthos Manager's Content Area.



Page Name: Welcome

Template: vtStandard/simple_page

Page Number: 100

Created: 26.05.2005, 17:06 Uhr by master

Modified: 24.01.2007, 12:37 Uhr by Masterbenutzer

Comment:

Properties

Sorting: by name ☐ descending order

Options: ☐ Protect page against moving and renaming
☐ Do not show this page in menus and sitemap
☐ Always show subordinate pages in Navigation Bar

Access Rights

	View	Change	Add/Remove
Owner: master	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Authorized Users: Alle	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Guests: anonymous	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

This picture illustrates the properties in Virtuos' Pro version. The “Access Rights” area of the Basic version looks slightly different (see the chapter “Managing Access Rights”).

3. Make your changes.

The meaning of the various specifications is described here:

<i>Page Name</i>	The internal name of the current page (limited to 35 characters). This the name used in the Header Area and the Navigation Bar, and can also be used to create links to the current page. The name is automatically determined by the page content when you create or edit the page content. Here, you can also set it manually. It will then remain fixed even if you make further changes to the text. To prevent unwanted renaming, you have the ability to “Protect page against moving and renaming,” as described below.
<i>Template</i>	The template on which the current page is based. The actual template name is preceded by the name of the package to which the template belongs, set off by a slash.
<i>Page Number</i>	The internal number of the current page. This number is automatically assigned when the page is created, and provides a unique identifier. It cannot be changed. The page number is used to define references that always remain valid, even if the page is renamed or moved.
<i>Created</i>	When and by whom the current page was created.
<i>Modified</i>	When and by whom the current page was last edited.
<i>Comment</i>	General comments about the current page. Comments can only be seen by users who are logged in, and can thus be used to store notes on such things as how to edit the page.

<i>Sorting</i>	Determines the order in which subpages of the current page are to be displayed. This affects the display in the Navigation Bar as well as in Structure mode and (unless otherwise specified in the template) Preview mode. A value of “manual” indicates that subpages should be sorted by position numbers set in Structure mode. The “random” option indicates that the subpages should appear in a different, random order each time the page is brought up.
<i>Protect page against moving and renaming</i>	Activating this option prevents the possibility of making changes that would make a website cease to function properly. It is intended for pages of central significance that are located in a fixed place in the tree structure and that need to have a fixed name.
<i>Do not show this page in menus and sitemap</i>	Activating this option makes the current page “invisible” in certain lists and menus. This means that it will not be shown in those lists and menus. This does not prevent a user from accessing and viewing the page through links to it. If you want to disable this as well, you must set the access rights accordingly (see the chapter “Managing Access Rights”). The availability of this option is determined by the applicable template.
<i>Always show subordinate pages in Navigation Bar</i>	When a page has more than 20 subpages, these are normally no longer shown in Virthos Manager's Navigation Bar. This means that the triangle in front of the page name can no longer be clicked. Activating this option allows subpages to displayed even if there are more than 20 of them.
<i>Access Rights</i>	The Basic and Pro versions differ in their behavior in this area. Since the Basic version does not allow for managing users, it only allows access rights to be set for guests; whereas in the Pro version, additional values can be set (but only by users with Administrator rights). More details are available in the chapter “Managing Access Rights.”

4. On the Toolbar, click [Save].

Your changes are made effective.

Changing the Sort Order

Whenever pages are listed (for example, in the Navigation Area or in Structure mode), they are always listed in a predetermined order. Pages are usually sorted by their creation date, but you can also set different sort criteria or set the order in which they are displayed manually.

You can find out how to have them sorted automatically in the topic “Modifying Page Specifications.” If you do not want them sorted automatically, but would rather set the order manually:

- 1. Bring up the page for which you would like to set the order in which its subpages are displayed.**
- 2. In the Mode Selector, click [Structure].**
The current page will appear in Structure mode, and all of its subordinate pages will be listed.
- 3. Set the order in which pages are to be displayed.**
Structure mode shows an input field to the left of each page that you can use to give each page a position number. The result might look like this:

<input type="checkbox"/>	Pos	Name	Template	Created	Modified
<input type="checkbox"/>	6	Company	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	5	Downloads	vtStandard/simple_page	24.01.2007, 12:52 Uhr	24.01.2007, 12:52 Uhr
<input type="checkbox"/>	2	Partners	vtStandard/simple_page	24.01.2007, 12:37 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	1	Products	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	4	Shop	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	3	Support	vtStandard/simple_page	24.01.2007, 12:54 Uhr	24.01.2007, 12:54 Uhr

6 Page(s)

- 4. Save your changes.**

Click on the column heading “Pos,” found above the input fields, or press the enter key while the cursor is in one of the input fields. The pages are then listed in the order that you have set:

<input type="checkbox"/>	Pos	Name	Template	Created	Modified
<input type="checkbox"/>	1	Products	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	2	Partners	vtStandard/simple_page	24.01.2007, 12:37 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	3	Support	vtStandard/simple_page	24.01.2007, 12:54 Uhr	24.01.2007, 12:54 Uhr
<input type="checkbox"/>	4	Shop	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr
<input type="checkbox"/>	5	Downloads	vtStandard/simple_page	24.01.2007, 12:52 Uhr	24.01.2007, 12:52 Uhr
<input type="checkbox"/>	6	Company	vtStandard/simple_page	24.01.2007, 12:53 Uhr	24.01.2007, 12:53 Uhr

6 Page(s)

Pages that do not have a position number (or the number “0”) are always shown at the end of the list. If more than one page has the same position number, these pages are listed in the order of their creation.

Please note!

The position numbers are saved even if you later switch to automatic sorting (for example, by name or creation date). To return to sorting by position number, just click the column heading “Pos” again in Structure mode, or, in Specifications mode, select “manually” from the “Sorting” list box. ¶

Removing Pages

If you no longer need a page, you can remove it from the page hierarchy:

1. **Bring up the page that you would like to remove.**

2. **In the Tool Bar, click [Put in Trash].**

The current page is put in the Trash Can.

You can also remove several pages at the same time by selecting them in Structure mode and then placing them in the Trash Can (see the topic “Moving Pages”).

If, on the other hand, you only want to prevent public display of a page temporarily, instead of following this procedure, you can just disable the “read” permission for visitors. More details are available in the chapter “Managing Access Rights.” You can read more about the Trash Can in the chapter “Introduction to Virthos Manager.”

Restoring Pages

If, after removing a page, you decide that you need it after all, you can restore it. This assumes that you have not emptied the Trash Can in the meantime. To restore a page:

1. **Bring up the Trash can.**

Click “Trash” in the Navigation Area.

2. **Change to Structure mode.**

Click [Structure] in the Mode Selector. You will see the contents of the Trash Can listed:

3. **Select the page that you want to restore.**

Check the box to the left of the page name.

4. **On the Toolbar, click [Cut].**

The current page is removed from the trash and placed on the Clipboard.

5. **Bring up the container page under which you want to place the restored page.**

6. **On the Toolbar, click [Paste].**

The page on the Clipboard then becomes a subpage of the current page, and can be edited as usual.

Exporting Pages

Virthos allows you to export selected portions of the page hierarchy and to import them again to a new location. This functionality lets you move pages from one Virthos installation to another (for example, from a local test environment to the web server) or create backups of portions of a website. Since

Virthos' import and export functions rely on an XML format, they also make it possible to transport Virthos data to other applications or to import data from other applications into Virthos.

Please note!

When exporting with Virthos, the entire content of pages is taken into consideration—not only the text, but also images and downloadable files. In Virthos Pro, moreover, only the content in the current language is exported, so that an export file created from a multilingual site does not represent a complete backup. ¶

To export pages from Virthos:

1. Bring up the page that you would like to export.

When exporting, the current page and any directly or indirectly subordinate pages are treated together. To export all the pages of a website, select the highest level of the Virthos hierarchy by clicking [Pages] in the Navigation Area.

2. Bring up the “Toolbox.”

Click the gear icon in the Mode Selector. The Content Area then shows you a list of additional functions.

3. Click “XML Export.”

The export process starts. This may take a while, especially if images or downloadable files are involved. The web browser then starts downloading the export file.

The export file is saved under the name “vtextextract.xml” by default. However, you can give it any name you want. To make the file format clear, we recommend that you use the “.xml” file extension.

Importing Pages

To import pages that you have exported from Virthos:

1. Bring up the page under which you want to place the pages that you are importing.

Imported pages are always inserted in the current position. To place them at the top of the Virthos hierarchy, click [Pages] at the very top of the Navigation Bar.

2. Next, set the access rights.

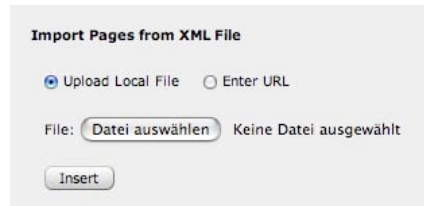
The imported pages will have the same access rights as the current page. It is therefore a good idea to ensure that the access rights of the current page are set the way you want (see the topic “Managing Access Rights”).

3. Bring up the “Toolbox.”

Click the gear icon in the Mode Selector. The Content Area then shows you a list of additional functions.

4. Click “XML Import.”

You will see a form listing various import options in the Content Area.



Virthos offers two import options. You may either

- upload a file from your hard drive, or
- enter the address (URL) of a file that has already been uploaded.

The second method is recommended for large files that could cause a timeout when you upload them using a web browser. In this case, you could use a FTP program to transfer the file to the web server and then enter the URL where it can be found.

5. Select the import file or enter the URL.

If you choose the “Upload Local File” option, you can then select a file from your hard drive by pressing the button. If you choose the “Enter URL” option, just enter the address of the file to be imported. This address must begin with `http://`.

6. Click [Insert].

The import process begins. If you are uploading a large file from your hard drive, the process may take a long time, and may cause your web server to time out. In this case, try using FTP or a similar means to transfer the file to your server; then enter its URL instead.

Please note!

When pages are imported, they may be assigned different page numbers from their original ones. This could cause problems when page numbers are used in references. References that have been assigned by authors using “styled” text variables are automatically updated by Virthos during the import process, but when there are references to page numbers within templates, these must be manually corrected in the source code of the template concerned. ¶

Editing Content

Users who enter text content in Virthos Manager have little influence on how that text is presented: The font, size, color and style are essentially determined by the underlying template. If a text placeholder is given the “styled” parameter (see the chapter “Introduction to VirthosTalk”), it is possible to manually format text to a certain extent. For example, words can be made bold or italicized. The following sections describe the various possibilities.

Formatting with Font Styles

To underline, make bold, or italicize a word, place special character strings around the word as follows. The leading string always begins with an underscore, and the terminating string always ends with an underscore. For example, if you were to write:

This is `_/the/_` opportunity to get to know one another better.

the word “the” is italicized:

HTML Source

This is `<i>the</i>` opportunity to get to know one another better.

Preview

This is *the* opportunity to get to know one another better.

Similarly, you can make individual words bold (underscore+exclamation point) or underlined (underscore+underscore). If you enter the following text:

We are always available on `_!Tuesdays!_`. Call us at `__(212) 754-1234__`.

appears as:

HTML Source

We are always available on `Tuesdays`. Call us at `<u>(212) 754-1234</u>`.

Preview

We are always available on **Tuesdays**. Call us at (212) 754-1234.

You can combine several font styles, for example, make part of an italicized string bold (or vice versa). This means that a string like

This is a `_!combination _/of multiple/_` formatting strings!_.

would appear like this when previewed:

This is a **combination of multiple formatting strings**.

If you want to combine several font styles directly one after the other, the underline string (if used) must always come first. Moreover, you should ensure that after a formatting string there is in the actual text no punctuation mark containing one of the formatting characters.

Applying CSS Styles

In addition to formatting with font styles, Virthos also allows logical markup within text. By “logical,” we refer to the purpose to which the text is put, instead of how it is formatted, as in the case of font, color, size, etc. The actual presentation depends on which browser you use to view the page and whether a CSS style has been applied to the element.

The following logical markups are available in Virthos:

<code>_# ... #_</code>	is used to denote source code (<code><code></code>)
<code>_: ... :_</code>	denotes keyboard or user-supplied data (<code><kbd></code>)
<code>_\$... \$_</code>	denotes variable names (<code><var></code>)

Through definitions in a CSS style sheet, you can determine how these elements will be displayed, in respect to their size, color, etc.

Bulleting Text

If you want to set off several text strings in an indented list with bullets, you can easily specify this with Virthos. You only need to place a hyphen (minus sign) before each item in the list:

- Monday
- Tuesday
- Wednesday

This is converted to:

HTML Source

```
<p><ul>
  <li>Monday</li>
  <li>Tuesday</li>
  <li>Wednesday</li>
</ul></p>
```

Preview

- Monday
- Tuesday
- Wednesday

Virthos places a “paragraph” tag before and after each item in the list (see the source text above). This means that the paragraph following the list does not require any special formatting or tags. It is therefore important to define a style for standard paragraphs in the CSS style sheet.

Internet and Email Addresses

When a placeholder is designated as “styled,” Internet addresses like “www.virthos.net” or email addresses like “info@virthos.net” are automatically converted to clickable hyperlinks. Moreover, email addresses are formulated in a special way that make them difficult for spam robots to recognize and read.

You can also make Internet and email addresses invisible by placing them “behind” another word. You do this by enclosing them within double parentheses:

For more information, write ((info@mycompany.com)) to us.

In this example, the email address “info@mycompany.com” will not be visible on a web page. Instead, the word “write,” which directly precedes the left parentheses, will be formatted as a link.

HTML Source

```
For more information,  
<a href="mailto:info@mycompany.com">write</a> to us.
```

Preview

For more information, write to us.

The same thing happens with Internet addresses, too. This means that a string like

To visit our store, click here ((www.mystore.com)).

becomes:

HTML Source

```
To visit our store, click  
<a href="http://www.mystore.com">here</a>.
```

Preview

To visit our store, click here.

In the case of links to FTP servers or SSL-protected web servers, the address must be complete, including the protocol name, for example:

```
ftp://ftp.mycompany.com or https://www.secureserver.com
```

The same is true for normal URL's that do not begin with `www.`:

```
http://shop.mycompany.com or http://virthos.net
```

Without the “http” at the beginning of this kind of address, Virthos would not recognize the string as an Internet address and thus would not show it as a hyperlink.

If you want to create a link out several words in a row, you must use a “no-break space” to separate the words. These special space characters can be created on a Macintosh computer by pressing the key combination Alt+space, or on a Windows computer by simultaneously pressing the Alt key and the number combination 0160. Not all web browsers allow you to enter no-break spaces. In case of doubt, you should use Safari on Mac OS X or Internet Explorer on Windows.

References to Other Virthos Pages

To refer to other Virthos pages, you may also use double parentheses. When doing this, precede the address with the abbreviation vt:

```
You can find more information here ((vt:234)).
```

This example illustrates a hyperlink to Virthos page 234. Instead of using a page number, you can also create a link using a path.

```
You can find more information here ((vt:Company/Addresses)).
```

The path specification shows where a page can be found in the Virthos hierarchy. In this example, it is the “Addresses” page under the “Company” page. Path specifications beginning with a slash (/) are always resolved relative to the start page. In the above example, the “Company” page must be located under the start page. For pages that are not subordinate to the start page, the page specification must begin with two slashes.

```
You can find the downloads available here ((vt://Internal/Downloads)).
```

This path points to the “Downloads” page under the “Internal” page. In this case, the “Internal” page can be found at the top of the Virthos page hierarchy. More information about page numbers and path specifications can be found in the “Entry Addresses” in Part 4 of this manual.

References to JavaScript Functions

Sometimes you may want a hyperlink to refer not simply to another web page, but to certain actions—for example, opening a new browser window with a specific size and position. If the template has enabled this kind of function, you can call it like this:

```
Click here ((js:functionname(parameters);)).
```

This source code becomes:

```
Click <a href="javascript:void()" ↵  
      onclick="functionname(parameters);">here</a>.
```

You should replace *functionname* and *parameters* with values derived from the actual JavaScript function. Be sure to remember the semicolon after the right parenthesis.

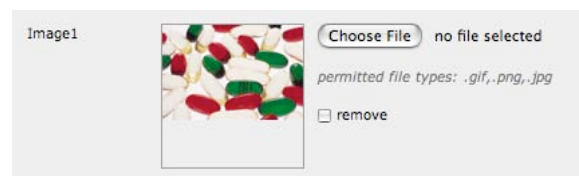
Uploading Images and Files

Many templates give you the ability to maintain not only text, but also images (or other files). In the form for creating a new page, you will then see an element such as this in these places:



The exact appearance will depend on which web browser you use to bring up the form. The process is the same, however. When you press the button, you will see a window where you can select a file from your hard drive. When you confirm your selection by pressing [OK], the selected file will be assigned to this element of the form. When you submit the form (by clicking [Save]), the file will be uploaded to the web server.

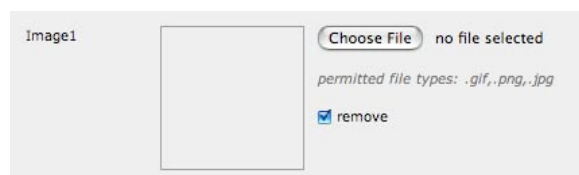
The next time you process the page, the form will show a similar element, the difference being that a thumbnail of the uploaded file is displayed instead.



In this case as well, you can use the button to select another file from your hard drive. The thumbnail is then replaced by an arrow that indicates that a new file is being uploaded.



When you submit the form, the current file is replaced by the one you are now uploading. If you want to remove the file without uploading another one, check the box next to the word “remove.”



Once you have checked the box, the thumbnail no longer appears, and when you submit the form, the current file is deleted from the web server.

Please note!

If you upload a large number of files or a single large file, submitting the form may take a while. While this is going on, you should not click any objects in the current web browser window or enter text in any other fields. If you determine at the end that a file that was uploaded is not displayed, or does not appear correctly, the file was probably too large. In this case, you should try to reduce the size of the file before uploading, perhaps by saving it in a different format, or—if you are uploading several files at once—try uploading them one at a time, by bringing up the editing form several times, and submitting it with only one file at a time. ¶

In System Settings, you can set an upper limit to size of files that may be uploaded. If a file exceeds this size, it will be uploaded to the server, but ignored by Virthos. This means that a file that is already present will then not be replaced on the page by the one that you just uploaded (see the topic “Security Settings” in the chapter “Managing System Settings.”

When you upload images and files, Virthos checks their format: Only images with .gif, .png or .jpg extensions are accepted. Other files that are uploaded may be in any format, as long as they do not present a threat to server security. For example, you cannot upload files with .php, html or .cgi extensions.

Managing System Settings

Virthos' behavior is controlled by several different means. To some extent, this is done through the configuration file *config.php*, described in the “Installation” chapter; to a large extent, it is controlled by system settings. System settings can be modified by “settings pages,” which can be brought up and edited in Virthos Manager just like regular pages. When you change the content of a settings page, the system settings are automatically changed.

The settings pages can be accessed by clicking [Settings] at the bottom of Virthos Manager's Navigation Bar. To do this, you must be logged in as the master user, or have Administrator rights. Your normal web pages are removed from the Navigation Bar, and you see the System Settings area instead. The settings are covered in several subpages, each of which can be invoked with a single click.

To change settings, you must edit the content of the corresponding settings page, as described in the topic “Changing Page Content” in the chapter “Managing Pages.” The meaning of the various systems settings is described below.

General Settings

General settings refer to the website in general. The following settings are available:

<i>Site Name</i>	A name that you can freely choose to represent your website. This name has two main functions: On the one hand, it is displayed in the browser's title bar when you access Virthos Manager (combined with the current user name). If you are running more than one instance of Virthos Manager in different windows, it makes it easier for you to know where you are. In addition, it is displayed in Virthos' built-in start page and—as long as Virthos is running in demo mode—in the demo announcement window. If no site name has been entered, Virthos displays the path to the <i>virthos.php</i> script.
<i>Logo</i>	The logo is displayed along with the site name on the built-in start page. As long as you have not substituted your own logo, the Virthos logo is used.
<i>Start Page</i>	The page you enter here is the one that will be displayed if you invoke the <i>virthos.php</i> script without any additional parameters, or if a guest user requests a page that does not exist. You can select any page that is displayed in the Navigation Bar directly under the “Pages” category, or a start page that you have built yourself that shows the site name and logo and that lists the top-level pages of the Virthos hierarchy. The built-in start page can be accessed in Virthos Manager by opening the “Pages” category in the Navigation Bar and then clicking on [Pages] again.

Database Settings

This settings page is available only in Virthos Pro. Its main purpose is to give an overview of the current mySQL database access information, as specified in the config.php file. However, this information cannot be changed in Virthos Manager. By clicking [Back Up Data], you can download a backup copy of the mySQL database tables used by Virthos to your hard drive (see the chapter “Backing up Data” in Part 1 of this manual).

Email Settings

This settings page can be used to control the way Virthos automatically sends email. More information on this topic is available in the chapter “Processing Forms” (Part 3 of this manual).

<i>Default Recipient Address</i>	This address is used to direct the messages created by Virthos. If this address is not provided, Virthos' automatic email function is disabled.
<i>Default Sender Address</i>	This email address is used as the sender of the messages created by Virthos. If this address is not provided, a default address is used.
<i>Maximum. File Size</i>	Maximum size of email attachments, in kilobytes. If a user uploads a file larger than this size in an email form, the message is not sent.

Licensing Settings

This settings page displays Virthos' licensing status, which you can change if you are logged in as the master user. If Virthos is running in demo mode for the current URL, you can use this page to order and enter a license key. Otherwise, you will see only a list of URL's for which license keys have already been entered. The current URL is shown in bold. Detailed steps for ordering and entering a license key are covered in Part 1 of this manual in the “Licensing” chapter.

Security Settings

This settings page can be used to control various aspects of security within Virthos.

Session Expiration Time If a logged-in user has not requested any pages for some time, Virthos will ask for a password to be re-entered after the specified period of time (in minutes). For security reasons, this interval should not be too long. Nor should it be too short, so as not to impede editing. A good compromise in most cases would be a value of 60 minutes.

Guest Access Without regard to the page-based access rights that are described in a later chapter, this area can be used to completely block guest access to Virthos pages. Virthos then requires a user name and password to be entered each time it is accessed. Locking guest access can be useful when there is basically or temporarily nothing available for the public to view, as during the construction of a website.

Maximum File Size This setting controls the maximum size of images and other files that can be added by authors. If an author tries to upload a file that is larger than what is specified here, Virthos will not accept it. In this way, you will avoid creating web pages that cannot be loaded by browsers because of excessively large images.

Language Settings

This settings page can be used to specify the languages in which authors may create and edit web pages (see the topic “Translating Page Content” in the chapter “Managing Pages”).

Languages

Active	Not Active
English (en)	Czech (cs)
German (de)	Danish (da)
French (fr)	Spanish (es)
	Hungarian (hu)
	Italian (it)
	Dutch (nl)
	Norwegian (no)
	Polish (pl)
	Portuguese (pt)
	Slovenian (sl)
	Swedish (sv)
	Turkish (tr)

Add Language Code

In the “Active” list are the languages that have already been set up. The “Not Active” list contains a list of frequently used languages. You can activate a language by selecting it from the list on the right and clicking on [<<]. Conversely, you can deactivate a language by selecting it from the list on the left and clicking on [>>].

To activate a language that is not present in the selection list, click [Add Language Code]. In the dialog window that appears, enter a two- or three-character code for the desired language, and then [OK]. You can choose any abbreviation that you find appropriate. However, we recommend that you use one of the codes in the ISO 639-1 standard. An ISO code has the advantage of being universally understood, and makes possible a secure exchange of content between various Virthos installations. Especially if you thinking of automatically directing visitors to your site to a language version based on their location, you should follow the ISO convention.

The sequence in which the active languages are arranged determines which version will be displayed to a visitor who either

- has not yet selected a language or
- who has selected a language for which there is no content yet.

An example will serve to illustrate this: Assume that English, German and French ("en" "de" and "fr") have been set up in this order. This would mean that visitors are first greeted in English. If a visitor selects "German," he would then see German content, but if he selected a page for which there was yet no German content, he would continue to see the English version. If the order of languages were "fr" "en" "de," he would then automatically see the French content, and he would only see the English version if there were no French version.

The order of appearance of the languages can be changed by selecting a language and using the arrows at the left to move it up or down.

Custom Settings

Virthos gives users the opportunity to enhance the system settings described above with their own user-defined settings. This functionality allows email addresses, telephone numbers or other content that appears frequently on many different pages to be managed from one central location. To take advantage of custom settings, you must first create a template for the settings page, and create a new settings page based on the template. Then, with the aid of the `vtGlobal` placeholder, you can integrate your custom settings into other templates.

Please note!

Another way of maintaining custom settings is to use the *config.php* configuration file. You can find more information on this in the “Configuration” chapter. ¶

Creating templates for custom settings pages

Each settings page, like any other page, is based on a template. The rules for creating one of these templates are the same as for regular templates (see Part 3 of this manual). However, only user-defined placeholders can be used in one of these templates (see the topic “Editable Text” in the chapter “Introduction to VirthosTalk”). Interchangeable images and downloadable files are not available on settings pages. Even the use of other VirthosTalk placeholders and directives is not usually effective.

You can give any name you like to the user placeholders that you incorporate into the template. The only point to keep in mind is that names of placeholders must be unique. If you use the same placeholders on several settings pages, changes made on one page will affect the content on another. This is true both for predefined system settings as well as for parameters used in the *config.php* configuration file (see the “Configuration” chapter in Part 1 of this manual). You may therefore not use any of the following names:

<code>baseURLFiles</code>	<code>baseURLMedia</code>	<code>baseURLResources</code>	<code>baseURLTemplates</code>	<code>build</code>
<code>firstLogin</code>	<code>languages</code>	<code>lastSessionCleanupID</code>	<code>licensedURL</code>	<code>licenseKey</code>
<code>locked</code>	<code>log</code>	<code>mailFileSize</code>	<code>mailRecipient</code>	<code>mailSender</code>
<code>masterPassword</code>	<code>masterUser</code>	<code>max_execution_time</code>	<code>mysqlDatabase</code>	<code>mysqlPassword</code>
<code>mysqlServer</code>	<code>mysqlTablePrefix</code>	<code>mysqlUser</code>	<code>pathToData</code>	<code>pathToDatabase</code>
<code>pathToFiles</code>	<code>pathToMedia</code>	<code>pathToTemplates</code>	<code>pathToVirthos</code>	<code>productName</code>
<code>refuseFileSuffix</code>	<code>sessionExpirationTime</code>	<code>showMaxSubpages</code>	<code>siteName</code>	<code>startPage</code>
<code>systemLanguage</code>	<code>version</code>			

You can also provide template variants for custom settings pages. If you have already set up a template by the name of *mysetup.html* for a settings page, you can build an editing form for this page with the name *mysetup.vtedit.html*. More information on this subject can be found in the chapter “Template Variants” in Part 3 of this manual.

Building Custom Settings Page

To create a page for your own settings:

1. In the Navigation Bar, bring up the “Custom Settings” page.

Please note that you must be logged in as the master user or as an Administrator in order to enter the “Settings” area.

2. **On the Toolbar, click [New Page].**
3. **Click “Show all packages.”**
You will then see a list of all available templates.
4. **Click the name of the template that you have selected for the settings page.**
You will see a form in which you can enter your settings.
5. **Fill in the form and click [Save] on the Toolbar.**
The settings page will be created and placed in the Navigation Bar under the “Custom Settings” page.
6. **If you wish, you can now rename the new page.**
The page name that is automatically created from the contents is usually not very meaningful. In the new page's specifications, you can give it any new name that you wish.

Your new settings page is now available. You can change the settings at any time later by accessing the page and clicking [Edit] in the Mode Selector.

Using Custom Settings

To incorporate a text string that is managed by a custom settings page into another template, use the VirthosTalk placeholder `vtGlobal`. For example, if the template on which the custom settings page is based contains the user placeholder

```
{{contactphone}}
```

you can use the expression

```
{{vtGlobal: contactphone}}
```

in any other template to pass the value that was assigned to the field “contactphone” when your settings page was created or modified.

Managing Access Rights

Virthos lets you control what any user can or cannot do. These capabilities are somewhat limited in Virthos Basic; however, in the Pro version, they are quite extensive. This chapter describes each of these capabilities in detail, and how to use them.

General Information About Access Rights

Access rights in Virthos are page-specific—for each page, you can decide who can see it, change it, or otherwise use it. All of these determinations can be made within the page specifications (see the topic “Modifying Page Specifications” in the chapter “Managing Pages”).

Whenever you create a new page, it first inherits the access rights of its parent page. This inheritance ends as soon as you manually change a page's access rights; they remain as you set them from that time forward.

In order to change access rights, you must be logged in either as the master user or a user with Administrator rights—or you must be the owner of the page concerned. The last two options are available only in Virthos Pro.

There are three types of rights defined within Virthos.

- | | |
|---------|---|
| Viewing | Users who have been granted the right to “view,” can see the contents of a page. If they do not have this right, they receive a warning notice—or in the case of guest users, they are redirected to Virthos' start page. In Virthos Manager's Navigation Bar and in all dynamic menus and sitemaps, users only see the pages that they have been granted the right to “view.” (This should be differentiated from the ability to make certain pages “invisible,” which causes them not to appear on most menus, although they can be accessed directly). |
| Modify | Users who have been granted the “modify” right for a given page can edit the contents of a page and change its specifications. Those who lack this right cannot make any changes using any of Virthos Manager's forms. Any attempt to use homemade forms to change the contents of a page will also fail. |
| Extend | Users who have the “extend” right for a given page can create new subpages for the page, or delete those that already exist. Lacking this permission, users cannot create any new pages, cut or delete any existing ones, or paste any content from the Clipboard. |

Access Rights in Virthos Basic

There is no user management facility in Virthos Basic. With Virthos Basic, you are either logged in as the master user, or you are a guest user—there is no further differentiation of users. Since the master user automatically has all possible rights, the page specifications can be used only to set the access rights that users who have not logged in (“guests”) have:

Access Rights		
View	<input checked="" type="radio"/> allowed	<input type="radio"/> not allowed
Change	<input type="radio"/> allowed	<input checked="" type="radio"/> not allowed
Add/Remove	<input type="radio"/> allowed	<input checked="" type="radio"/> not allowed

The meaning of these three access rights are described in the topic “General Information About Access Rights.” Here are just a few more remarks concerning guest users:

- If you set “View” to “not allowed,” you are making the current page not viewable for the public. As a result, the Navigation Bar will show a “locked” icon to the left of the page name, and guest users will no longer be able to access the page. Even if a user had made a note of the page's exact URL, Virthos would no longer display the page, and would redirect the user to the start page instead.
- Granting the “modify” right to guest users would most likely not make sense, because you would then be giving this right to people whom you do not know. But if this really is your intention, you would need to create the corresponding editing form or make available a link to the “vtedit” method, so that guests could actually use the privilege.
- On the other hand, the “extend” right for guest makes a lot of sense, because guest users should be able to create new entries, as in a discussion forum, a guest book or a “blackboard.” In each of these cases, the appropriate forms would need to be made available, so that guests could actually take advantage of this right.

Access Rights in Virthos Pro

With Virthos Pro, you have the ability to create an unlimited number of users or user groups in order to grant graduated access rights (see the chapter “Managing Users”). The Page Specifications thus allow access rights to be defined not only for guests, but also for two other classes of users:

Access Rights		View	Change	Add/Remove
Owner	Editors	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Authorized Users	All	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Guests	anonymous	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

These three classes of users are defined this way:

- Owners* The owner of a page has basically unlimited rights with regard to the page. The check marks next to “View,” “Change” and “Add/Remove” can therefore not be removed. The user who creates a new page automatically becomes its owner. Afterwards, you can make any other user or user group an owner, too. Please note however: If you change the owner of a page, you yourself may not be able to change its access rights later.
- Authorized Users* In addition to the owner, you can establish authorized users for a page, and give them (which can also be user groups) their own access rights. If you choose not to use this feature, you can set Authorized Users to “<none>.” When a new page is created, the authorized user of the parent page automatically becomes an authorized user of the subordinate page.
- Guests* This line is used to set the access rights for users who are not logged in—meaning the public. This setting is the same as in Virthos Basic, described above in the topic “Access Rights in Virthos Basic.”

It is advisable to define only groups, and not individual users, as owners and authorized users. Then, if the composition of your groups changes, then you would need only to remove an old user or add a new one to a group. Otherwise, you would find yourself having to change the access rights for many pages individually instead.

Managing Users

Virthos' Pro version lets you create as many user names as necessary for your staff who need to log in to the system. You can use these user or login names to determine precisely which staff members can view or modify a given page. This chapter describes this process in detail.

Virthos Basic does not provide for user management, since it distinguishes only between the master user and others. Anyone who knows the master password can access any of Virthos' pages and modify or delete them. The previous chapter explains this in detail.

General Information About Users and Groups

In managing access rights, Virthos distinguishes between “users” and “groups.” A user is a person who uses a specific login name to access Virthos. If two or three people log in with the same name, Virthos treats them as one and the same. Conversely, one person may log in to Virthos with two different names and be considered by Virthos to be two different users.

Users can be associated in user groups. A group has an assigned name, but it is not possible to log in to Virthos with this name. The purpose of establishing groups is to simplify the management of user rights, so that several persons can perform similar duties within Virthos and can thus be given the same access rights.

A user can be assigned to as many groups as needed. If these groups have different access rights, users are given the maximum rights. If a user is assigned to both the “News” and “Products” groups, he can modify both the pages assigned to the “News” group and those assigned to the “Products” group.

Virthos automatically creates and maintains the “All” group. All users are automatically members of this group and cannot be removed from it. It is up to you whether to define access rights for the “All” group. It is made available to you in case there are rights that you would like to grant to all logged-in users, irrespective of their other group affiliations.

Accessing User Management

Each Virthos user or user group has its own system settings page that can be used to define the properties of the user or group. These system pages are located in their own category in Virthos Manager's Navigation Bar (see the chapter "Introduction to Virthos Manager"). To access User Management:

1. **In Virthos Manager's Navigation Bar, open the "Users" category.**

Click the "Users" button at the bottom left of the Navigation Bar.



The button is shifted to the top, and a list of available user groups is displayed. Virthos Manager's Content Area also shows a list of available groups. Until you create groups of your own, you will only see the "All" group.

2. **Access a group or a user.**

Click the name of a group to see the group's properties and a list of its members. You may also click the gray triangle to the left of the group name in the Navigation Bar to see which users belong to it. Clicking on an entry in the list will bring up the user's properties in the Content Area.

Creating a New User

Virthos lets you set up any number of users with their own login names and passwords. To create a new user:

1. **Bring up User Management.**

2. **On the Toolbar, click [New User].**

The Content Area will show a form in which you can enter the user name and password.

3. **Fill out the form.**

The meaning of the individual fields is described below:

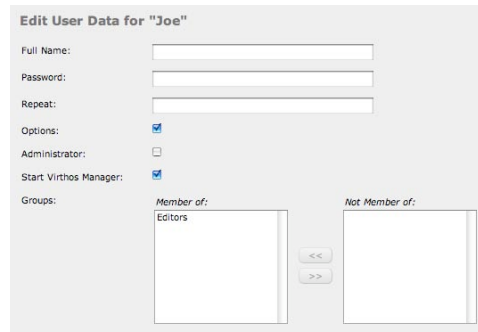
User Enter the name that the user will use to log in to Virthos. The name may consist of letters, digits, underscores and hyphens, and must not exceed 20 characters. Make sure that the name has not already been assigned, since otherwise, you will see an error message when you try to save it. The names "master," "system" and "guest" are reserved for internal purposes and also cannot be used.

Password The password to be assigned to the user name. It can be up to 40 characters long, and for security reasons, should include uppercase and lowercase letters, as well as digits and punctuation marks. The characters are not displayed; instead you see only a row of asterisks.

Repeat Enter the password again. Virthos verifies that this entry is the same as the password that you previously entered. If not for this check, if you had mistyped the password, the new user might not be able to log in.

4. On the Toolbar, click [Save].

The form is submitted, and the new user is created. The Content Area now displays a form where additional properties can be specified for the user.



5. Fill out the form.

The meaning of the individual fields is as follows:

Full Name You can use this field to enter the full name of the user, or a short description. This information is for your own purposes only, and has no effect on the user's access rights within Virthos.

Administrator If the user is to have extended access rights, check the "Administrator" box. The user can then have access to system settings and can manage users and access rights. A user with Administrator rights is exempt from any limitations that you might set up. This user can view, edit and delete any page.

Groups The groups to which the user belongs, or does not belong. To assign a user to a group, select the desired group from the list at the right and click on [<<]. To undo a group assignment, select the group from the list at the left and then click on [>>].

Activated Check this box to allow a user to log in to Virthos. This option is also useful if you want to prevent a user from accessing the system, but do not yet want to delete the user information. In that case, you can just clear the box.

6. On the Toolbar, click [Save].

The form is submitted, and the user properties are altered according to your specifications.

The new user is then available. As long as you have enabled logins, your staff can immediately start using the name and password that you have set up to log in to Virthos.

Creating a New Group

To create a new user group:

1. Bring up User Management.

2. On the Toolbar, click [New Group].

The Content Area will show you a form where you can enter a group name.

3. Enter the group name that you have selected.

The name may consist of letters, digits, underscores and hyphens, and must not exceed 20 characters. Make sure that the group name has not already been assigned, since otherwise, you will see an error message when you try to save the form.

4. On the Toolbar, click [Save].

The form is submitted, and the new user group is created. The Content Area now displays a form where additional properties can be specified for the group.

The screenshot shows a web form titled "Group 'Editors'". It contains a "Remarks:" label followed by a text input field. Below this, there is a "Users:" label. To the right of "Users:" are two list boxes: "Members:" and "Non-Members:". The "Members:" list box contains the name "Joe". Between the two list boxes are two buttons: "<<" and ">>".

5. Fill out the form.

The individual fields have the following meaning:

Remarks You can enter information about the purpose of the new group here. When you have a great many groups, you will find that these tips make it easier to assign users.

Users The users who are to be members or non-members of the group. To assign a user to a group, select the desired user name from the list at the right and click on [<<]. To remove a user from the group, select the user's name from the list at the left and click on [>>].

6. On the Toolbar, click [Save].

The form is submitted, and the group properties are altered in accordance with your specifications.

The new group is then made available, and you can now define access rights for it.

Changing User and Group Properties

With the exception of the name, you can later change any specifications made when you created a user or a group. In particular, you can assign users to new groups at any time, or remove users from groups. To do this:

- 1. Bring up User Management.**
- 2. Access the desired user or group.**
Click the name in the Navigation Bar. The properties of the user or the group are shown in the Content Area.
- 3. In the Mode Selector, click [Edit].**
You will see a form allowing you to change user or group properties in the Content Area.
- 4. Change the information you want.**
You will find a description of the entry fields under the topics “Creating a New User” and “Creating a New Group.”
- 5. On the Toolbar, click [Save].**
The form is submitted, and the your changes are saved.

Deleting Users and Groups

When you no longer need a particular user or group, you can remove it with Virthos' User Management. In contrast to the process of deleting web pages, there is no Trash Can which can be used to restore deleted objects. Deletion of users and groups is final.

Please note!

After you delete a user or group, you can immediately re-create a user or group with the same name. However, the access rights connected to the deleted object cannot be restored. An accidental deletion can result in considerable correction work where a large site is concerned. For this reason, we recommend that you first remove the login privilege for a user before deleting the user. Before you delete a group, you should first remove all its members and add a note that the group will shortly be deleted. If this does not result in any unforeseen difficulties over a period of time, you can assume that there is no harm in later deleting the user or group. ¶

To permanently delete a user or group:

1. **Bring up User Management.**
2. **Access the desired user or group.**
Click the name in the Navigation Bar. The properties of the user or the group are shown in the Content Area.
3. **On the Toolbar, click [Delete].**
A dialog window appears to allow you to confirm your intention to perform the deletion.
4. **Click [OK] in the dialog window.**
The user or group is then permanently deleted.

If the deleted user or group was either an owner or an authorized user of a page, the page specifications will show the word “unknown” instead of the user's name or the group's name.

PART 3

Templates

Creating and Installing Templates

Like almost all other content management systems, Virthos is based on templates. A template is a kind of layout model that describes how text and images can be accommodated on a page and that allows these to be processed through a graphic interface. On a Virthos-based website, each page is based on a template. You must select a template from a list when creating the page. The same template can be used for any number of pages. If you modify a template later, the change affects the appearance of all the pages based on it. Since the text and images of the individual pages are not affected, it is easy to use this method to apply a new design to a large number of pages within an Internet site.

A Virthos template is normally a simple HTML file, yet with one important difference. In addition to the usual HTML elements, a template can also include placeholders and directives. Placeholders are used to indicate where text and images should be inserted; you can use directives to specify, for example, that a given HTML block should be repeated multiple times. The language used to formulate these directives is called VirthosTalk. It is thoroughly documented in its own Reference Guide. The following chapters will introduce you the most common VirthosTalk language elements.

Design Rules

Since a Virthos template consists basically of HTML code, you can create a template with the same tools you usually use to design web pages—for example, with an HTML source code editor or a graphic web design program like GoLive or Dreamweaver. Moreover, there are no limits placed on your ability to use various tools when creating your design: You can use any technique known to the World Wide Web, whether XHTML, DHTML, CSS—or whatever the future may bring. There are three points to keep in mind when designing templates, however:

- All resources must be located in subdirectories.
- The file must be saved in the ISO Latin 1 character set.
- Any automatic URL coding in your web design program must be disabled.

The details of these three points are discussed below.

All Resources in Subdirectories

All images, style sheets, scripts and other resources pointed to from the template must be located in subdirectories of the template's directory. This means that they cannot be in the same directory as the template itself, and they cannot be higher in the hierarchy or at the same level in other branches. The path to a resource cannot begin with two dots, and it must contain at least one slash.

Correct: ``
Incorrect: ``
Incorrect: ``

If you fail to follow this rule, any resource used by a template will not be found when the page is accessed. The only exception to this strict rule is that you may reference a resource through an absolute path that begins with a slash, for example:

Correct: ``

References of this sort make it difficult to use templates in local test environments and should therefore be used only in special circumstances.

Please note!

VirthosTalk placeholders cannot be used within resource paths, including the resource file name. If you want to dynamically “derive” the name of a resource, you must have previously stored the entire path in a variable, using the `vtSet` directive, and insert it at the desired location using a `vtGet` placeholder (see the topic “Variables” in the chapter “Introduction to VirthosTalk”). ¶

Coding in ISO Latin 1

An HTML file that is used for a Virthos template must be created with the ISO Latin 1 character set and must also be designated as such. The following line should be included in the HTML header (`<head>` section):

```
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
```

Without this directive, many browsers might not correctly interpret certain characters that might be used by your authors.

No Automatic URL Coding

You must disable automatic coding of special characters in URL's, a feature of many web design programs. Otherwise, many hyperlinks and resource paths defined with VirthosTalk placeholders will not function correctly. If, for example, you enter a placeholder like `{{vtLink}}` in GoLive, if automatic URL coding is turned on, the source code will be converted to `%7b%7bvtLink%7d%7d`. This character string would then not be recognized as a placeholder by Virthos. Please refer to your web design program's documentation to find out how to disable automatic URL coding. For example, in GoLive this option is called “HH-Conversion” and can be found in Settings under “General > URL Management.”

Naming Rules

For naming templates, the same basic rules apply as for other HTML files. This means that names can consist of only the following characters:

- the letters a through z
- the digits 0 through 9
- the hyphen or minus sign
- the underscore character (" _ ")

Foreign characters, such as the German umlauts (ä, ö, ü), other accented letters and ligatures (é, à, œ etc.) and special characters of any kind should not be used, since doing so could lead to problems of various sorts on many web servers and browsers.

In addition to these general rules for naming Virthos templates, the following rules also apply:

- The name of the file must end in .html, so that Virthos will recognize it as a template. Other file extensions (such as .htm) are not allowed for templates.
- Names of template files should consist entirely of lowercase characters, since using uppercase letters could cause problems on many servers.
- The file name may not start with a hyphen, and can start with an underscore only in limited circumstances (see the chapter “Template Variants”).
- Periods have a special meaning in template file names. They serve not only to set off the file extension, but also to indicate template variants (see the chapter “Template Variants”). For these reasons, periods cannot be part of the actual file name.

You can find more information concerning the naming of templates, including accommodating different language versions in the topic “Template Variants.”

Installing Templates

In order for an HTML file to be used as a template in Virthos, it must be located in the “templates” subdirectory. In a standard installation, this “templates” directory can be found beneath the Virthos main directory—in other words, in the directory that contains *virthos.php*.

Templates cannot be located directly in the “templates” directory—they must be in one of its subdirectories, which in this context are called “template packages” or simply “packages.” A package may contain any number of templates, and will usually contain all the resources, such as images, style sheets, scripts, that the templates use.

Please note!

When Virthos is installed on a web server that uses a Unix operating system (including Linux and Mac OS X), all subdirectories and files found in the “templates” directory must be given a general “read” permission. Otherwise, Virthos will not be able to access the templates. ¶

Post-Implementation Modifications

A page that was built using a template always remains associated with that template in Virthos. The template is used to construct the page not only when it is created, but each time the page is accessed later. Changes made to a template immediately affect all the pages that are based on it.

This automatic updating may sometimes be undesirable, as when you are just testing a new template design for an existing website, and are not yet ready to publish it. In cases like this, you may want to create a template variant for the new design, and to update the standard template only when the new design is ready to be released. Details can be found in the next chapter.

Template Variants

Virthos lets you provide variants of each template that make it possible to display the same page content in different ways. For example, you can create a “printer-friendly” version of each template, containing only text and images and eliminating navigation menus, background designs or other elements that would detract from a printed copy. You can also use variants to try out a new design without making it public, to make different language versions appear differently, or to replace Virthos' standard forms for creating and editing web pages with your own forms.

This chapter will often make use of the concepts “template,” “method” and “page type,” all of which relate to each other, but have different meanings. To avoid any misunderstandings as you continue to read, you should try to understand exactly how these terms are used.

Template

The word “template” designates a file (normally an HTML file) that Virthos uses to present content. To make it quite clear that we are talking about concrete files, we will sometimes use the term “template file,” which has the same meaning as “template.”

Page Type

The term “page type” designates the type assigned to a page created with Virthos. The page type is the same as the name of a page's underlying template, but without the ending “.html.” A page based on the “article.html” template has “article” as its type.

Method

The term “method” in Virthos is used to refer to an action that processes or displays the contents of a page. A template can be thought of as the result of using a particular method to describe a page type. The template name is composed of the page type, the method name, and the file extension. When reference is made to the default method for a page type, the method name is omitted. When a global method is involved, the page type is omitted. The following topics give a further explanation.

User Methods

Suppose you have created a template called “article.html” and you would now like to create a variant that is optimized for printing. You could perhaps call the variant “article.print.html.” This means that you are extending the name of the default template by adding a period and an additional qualifier that designates the variant. The additional qualifier used in this name is also the name of the method that is defined by the template.

You can choose any method name that you like; however, the same rules apply as for template names (see the topic “Naming Rules” in the chapter “Creating and Installing Templates”). The name may consist only of letters, digits, hyphens and underscores. You should also be aware that certain methods have a special meaning for Virthos. If you create variants for these “system methods,” you will change

Virthos' behavior (see the topic "System Methods"). Unless you are fully aware of what you are doing, do not use any method names that begin with "vt."

To access a page using a specified method, you must add the file name suffix of the variant to the URL using the parameter "-met." For example, using this URL:

```
http://.../virthos.php?-pg=112&-met=print
```

you would bring up page 112 with the "print" method. That means that if this page had a type of "news" (based on the "news.html" template), Virthos would then look for a template file named "news.print.html" and would draw on that for its presentation, assuming that the file existed. If you were to omit the -met parameter, Virthos would look for the default template for the page in question.

If you were to create a hyperlink using the VirthosTalk placeholder {{vtLink}}, you could also specify a method, for example:

```
{{vtLink: -met="print"}}
```

When used to access a page, this placeholder would be replaced by the name of the page concerned followed by the "print" method.

System Methods

Certain method names have a predefined function for Virthos. The names of these "system methods" always begin with "vt" and are used to build the Virthos Manager interface. By providing your own templates with these method names, you could tailor the user interface to your own requirements.

The two most important system methods are:

- | | |
|---------------|--|
| vtedit | This method creates the form that you use to edit the content of a page. Any time a user clicks [Edit] in Virthos Manager's Mode Selector, he invokes this method. |
| vtnew | This method brings up a form for building a new web page. It comes into play when a user clicks [New Page] in Virthos Manager's Toolbox and selects a template. |

If you have already created a template named "product.html," you can then supplement it with an editing form called "product.vtedit.html." If a user then clicks [Edit] on a page of type "article," instead of seeing Virthos' default editing form, he will then see the one specified by "product.vtedit.html." Similarly, you could use your own form for building a new "product" page by using a template called "product.vtnew.html," thus overriding the default form.

When building your own forms, you should pay particular attention to assigning the correct names to form fields and specifying the correct "action" attribute. Otherwise, your form may sometimes not function as desired. More information about this subject is available in the chapter "Processing Forms."

Global Methods

You can use Virthos to construct templates that are not related to a specific page type or default template, but instead have a global character. These “global methods” can be accessed for pages of any type. A typical use for a global method would be a form that visitor uses to send an email to a friend telling him about the page that he is currently viewing.

Any method can be designated as a global method if the first part of the file name (the page type specification) is replaced by an underscore. A file named

`_recommend.html`

would be used to define a global method with the name “recommend.” This method could be invoked in the same way as a normal template variant, for example, like this:

```
{{vtLink: -met="recommend"}}
```

Because global methods are applicable to all pages, you should only use them with placeholders whose names begin with “vt” (for example, `vtName` or `vtID`) or those that occur often in all page types.

Language Variants

Virthos' Pro version lets you build template variants for different languages. This makes it easy to make available different language versions for menus, captions, etc. To build a template variant for a given language, the appropriate language code must appear in the file name. So the German variant for the “news.html” template would appear as “news_de.html,” and the French version as “news_fr.html,” and so on. The language code must always be appended to the main part of the template name with an underscore.

You can also create different language versions of templates for which there are methods defined. So the German variant for the “news.print.html” template would appear as “news.print_de.html,” and the French version as “news.print_fr.html,” and so on. The same holds true for global methods: The German version of the previous example “_recommend.html” would thus be named “_recommend_de.html.”

The appropriate language version of a template is automatically invoked, based on the currently selected language (see the “Entry Addresses” chapter in Part 4 of this manual). If there is no variant available for the current language, Virthos always goes back to the default template for the current page—meaning the template file that does not have a language identifier as part of its name.

The language codes used in template names must correspond to those that were used when establishing the Virthos system settings (see the chapter “Managing System Settings”).

Introduction to VirthosTalk

The fundamental difference between a Virthos template and a regular HTML file is that a template can use not only standard HTML tags, but also placeholders and directives that Virthos subsequently interprets. These Virthos-specific elements make up their own, language, known as “VirthosTalk.” The possibilities opened up when a template designer uses VirthosTalk are similar to those offered by server-side scripting languages like PHP or Perl. However, VirthosTalk is significantly easier to learn.

VirthosTalk is a language that is interpreted on the web server, and not on the browser. This means that all VirthosTalk placeholders are already replaced by the proper text or image content when they appear in the browser. A visitor to your site will not find any reference to the original VirthosTalk elements in the template when he views the source code. This also means that you cannot use VirthosTalk to trigger any more actions once the web page has been loaded by the browser. You must make use of the current JavaScript technology to accomplish this.

This chapter presents an introduction to VirthosTalk from the point of view of the user. Anyone who has previously worked with scripting languages and who is looking for a systematic introduction should refer to the VirthosTalk Reference Manual, available for download at www.virthos.net. The Reference Manual also describes placeholders and directives that are not described in this chapter, or are just mentioned briefly.

General Syntax Rules

The basic rule of VirthosTalk is: Anything to be interpreted by Virthos should be enclosed in double braces.

any HTML code `{{a VirthosTalk element}}` *any HTML code*

Virthos ignores anything outside the double braces. Whether they are HTML or JavaScript code, style sheet definitions or XML elements, the parts of the code outside the double braces may at the most be bypassed or repeated; however, the code itself stays unchanged. There are no exceptions to this rule.

Within the double braces, the first (and often only) element will be either the name of a placeholder or a directive:

`{{placeholdername}}`

for example:

```
{{Headline}}
{{vtLink}}
{{vtLoop}}
```

The important thing to remember is that the placeholder name appears directly after the left double braces, and should not be separated from them by a space.

Correct: `{{Headline}}`
 Incorrect: `{{ Headline }}`

Parameters

Often a further specification (called a parameter) may be needed. A parameter must be separated from the placeholder name by a colon:

`{{placeholdername: further specification}}`

for example:

```
{{Headline: styled}}
{{vtLink: 123}}
{{vtLoop: article}}
```

If several additional specifications are required, these are separated from each other by commas, for example:

`{{vtLink: 123, print}}`

When there are more than two parameters (and sometimes if there are two), you should use a convention that gives each parameter a unique name:

`{{placeholdername: Parameter1="Wert1", Parameter2="Wert2" ...}}`

for example:

```
{{vtLink: -pg="123", -met="print"}}
```

Instead of using double quotes, you can also use single quotes, but within a placeholder, you must opt for one method or the other:

Correct: `{{vtLink: -pg='123', -met='print'}}`

Incorrect: `{{vtLink: -pg='123', -met="print"}}`

Placeholders in Parameters

Inside parameters, you can, in many cases, use other placeholders. In the case of such nesting, the rule is that the inner placeholders are written only with single braces, for example:

```
{{vtLink: -pg='{pagenumber}' }}
```

When positional parameters are specified with placeholders, you should be careful not to place the inner and outer closing braces next to each other, but instead separate them with at least one space:

Correct: `{{vtLink: {pagenumber} }}`

Incorrect: `{{vtLink: {pagenumber}}}`

Directives

VirthosTalk elements that serve as directives and not as placeholders for other content can be written in the form of HTML comments to make them easier to recognize, for example:

```
<!--{{vtLoop}}-->
```

Please make sure that there are *no spaces* between the HTML comment indicators and the braces; otherwise the template may not function correctly.

Editable Text

To indicate a place in the template where editable text can be entered, you must insert a “user placeholder” into the HTML code—that is, a placeholder with a name of your choosing. Possible names for such user placeholders could be:

```
{{Headline}}
{{Body}}
{{Caption}}
```

You can choose any name you like, as long as it adheres to the following rules:

- The name may consist only of letters of the alphabet, the digits 0 through 9, hyphens and underscores. Other special characters are not permitted—nor are accented letters.
- The first character of the name must be alphabetic.
- The name may not start with “vt,” since this prefix is reserved for predefined VirthosTalk placeholders.

Placeholder names are not case-sensitive. `{{Text}}`, `{{text}}` and `{{TEXT}}` all refer to the same placeholder.

When you create a new page with Virthos, for each placeholder, there is an entry field set up where an author can enter text. When the page is accessed in a browser, each placeholder is replaced by the text that was entered when the page was edited.

Line Breaks and Special Characters

In order for text to be displayed correctly in a browser, Virthos replaces all line breaks with HTML break tags (`
`), and replaces accented letters and special characters with so-called HTML entities. For example, in German, “ä” is replaced by `ä`, and “ß” with `ß`; If this automatic conversion is not wanted, the placeholder name must contain the descriptor “raw,” as in

```
{{Headline: raw}}
```

This syntax is often required when the placeholder appears inside a multiline field on a form (`<textarea>`) or if the text is to be used inside a JavaScript function.

Marking up Text

Virthos can also convert certain types of markup to HTML styles. An author can use these to make parts of text bold or italicized. To allow these added distinctions, the placeholder name must contain the “styled” descriptor:

```
{{Body: styled}}
```

The types of markup that are available and how these are converted to HTML are thoroughly described in the chapter “Editing Text” in Part 2.

Interchangeable Images

To set a location in a template where an interchangeable image can appear, the URL must contain a “vtMedia” placeholder, for example:

```

```

or

```

```

The descriptor following `vtMedia` is the name that Virthos assigns to the picture. This name can be freely chosen, but the same rules apply as for naming user placeholders (see previous topic).

When you create a new page with Virthos, a location on the form where you can select the image to be uploaded is set up for each `vtMedia` placeholder in the selected template. When the page is displayed, each placeholder will be replaced by the image's actual URL. If there is no image to be uploaded, the URL of an invisible picture (*clear.gif*) in the system/resources directory is referenced instead.

Uploaded images are managed by Virthos in the *data/media* directory. Image files are automatically assigned unique names so that there can be no conflict even for two different `vtMedia` placeholders in a template you upload files with the same name but with different content.

Images as Backgrounds and in Style Sheets

Since `vtMedia` placeholders are replaced by plain URL's, you can use them not only in `` elements, but also to specify a background for a page or in style sheets.

Example for Using an Image as a Page Background

```
<body background="{{vtMedia:Background1}}">
```

Example for Using an Image in a Style Sheet

```
<div style="background-image:url('{{vtMedia:Background1}}');">
```

Size of Image Files

When using a `vtMedia` placeholder, you must decide to what extent the layout can be determined by the image's actual size and how much freedom you want to give to your authors to set the size. In the following example, the image will always appear in its original size:

```

```

and in this example, it will always appear with a fixed width of 145 pixels:

```

```

This means that in the browser, the image may be enlarged or reduced. Here, the predetermined width is also made part of the image name as a reminder to authors ("Photo_145px"). Specifying both the width and height is seldom effective, since images may be distorted if they are incompatible with the rest of the page content.

In any case, you should make the person responsible for your site's content aware of the consequences of using images that are too big or too small. In Virthos' system settings, under "Security," you also have the ability to specify a maximum file size for images. If an author uploads a picture file that exceeds this size, Virthos will ignore it. This will prevent the situation where a page cannot be loaded because of an excessively large image.

Interchangeable Downloadable Files

In addition to text and images, Virthos also lets you maintain downloadable files—that is, PDF, Word or other files that are not displayed in the browser, but can only be downloaded by the reader. To place a reference to one of these files in a template, the URL must include a `vtFile` placeholder:

```
<a href="{{vtFile:File2}}">download</a>
```

The name following the colon is the name under which Virthos manages the file. This name, too, can be freely chosen, but the same rules again apply as for naming user placeholders (see the topic “Editable Text”).

When you create a new page with Virthos, an object is created on the form to allow you to specify the file to be uploaded for each `vtFile` placeholder in the selected template. When the page is displayed, each placeholder will be replaced by the URL of the file that has been uploaded. In contrast to the `vtMedia` placeholder, `vtFile` renders a null string if no file has yet been uploaded to the server.

Managing Downloadable Files

Downloadable files are managed by Virthos in the *data/files* directory. In contrast to image files, downloadable files keep their original names, since these may often have meaningful names. When there are several `vtFile` placeholders in a template, authors must ensure that two files with the same name are not uploaded.

Revealing File Names

If you would not like to reveal the entire path, but only the name of a file that you have uploaded, you can achieve this result using the `vtFilename` placeholder.

```
<a href="{{vtFile:File2}}">{{vtFilename:File2}} </a>
```

After `vtFilename`, the same name as in the `vtFile` placeholder is used—“File2” in the above example.

Just as for images, you can set a maximum size for downloadable files. This value can be set in Virthos' system settings under “Security.” If an author uploads a file to the server that exceeds the maximum, Virthos will ignore it.

Dynamic Lists

Virthos lets you build dynamic lists that can be easily maintained by authors. A list of telephone numbers could look like this in HTML:

```
<h1>Phonelist:</h1>
<p>
  <!--{{vtLoop: Person}}-->
    <b>{{Name}}</b>, {{Phone}}<br>
  <!--{{vtEndLoop}}-->
</p>
```

The major factor in this example is the `vtLoop` directive in the third line. It indicates that the HTML block between `vtLoop` and `vtEndLoop` should be repeated as many times as there are entries in the list. The “Person” descriptor after `vtLoop` can be any term; it establishes the “type” of list entry and helps to distinguish the list from other lists that may eventually appear on the same page.

Virthos treats each `vtLoop` list entry as its own page. This means that the list items appear in Virthos Manager's Navigation Bar as subordinate pages of the page containing the list. This allows a list to be processed with the same tools that are available for managing pages:

To...	... do this:
add a list item	On the Toolbar, click [New Page].
remove a list item	select the item in the Navigation Bar and click [Put in Trash] on the Toolbar
change the order of list items	click [Structure] in the Mode Selector and set the position number, or click [Specifications] in the Mode Selector and select another automatic sort sequence.

Aids for Authors

Since it could be inconvenient at times to edit a list using Virthos' standard tools, you should simplify this task for your authors—specifically by providing small buttons located next to the list items that can be used to delete or edit them. You can add these buttons with the help of the `vtButton` placeholder.

```
<h1>Phonelist:</h1>
<p>
  <!--{{vtLoop: Person}}-->
    <b>{{Name}}</b>, {{Phone}}
    {{vtButton:edit}} {{vtButton:trash}}<br>
  <!--{{vtEndLoop}}-->
</p>
```

The `vtButton` placeholder is replaced by a small button that can be used for the indicated function: (edit = edit item, `trash` = put in trash). These buttons are available only for authors—Virthos does not show them to visitors to your site.

Nested Lists

You can nest two or more `vtLoop` lists. In that case, you might want to provide buttons for adding and sorting items, as well.

```
<h1>Phonelist: {{vtButton:new,Department}} {{vtButton:sort}}</h1>
<!--{{vtLoop: Department}}-->
  <h2>{{DepartmentName}}
    {{vtButton:new,Person}} {{vtButton:sort}}</h2>
  <p>
    <!--{{vtLoop: Person}}-->
      <b>{{Name}}</b>, {{Phone}}
      {{vtButton:edit}} {{vtButton:trash}}<br>
    <!--{{vtEndLoop}}-->
  </p>
<!--{{vtEndLoop}}-->
```

In this example, there is an “outer” list with department names, and for each of these, an “inner” list with the names of the people in the department. Each list can be sorted independently of the others. The second parameter following `new` is used within `vtButton` to show to *which list* an item should be added. Here, you must use the type designation in the corresponding `vtLoop` directive.

Conditional Layout Elements

It is sometimes desirable to display certain elements of a template only if specific conditions have been fulfilled, such as when a placeholder actually has a value assigned to it. An example might look like this:

```
<p><strong>Note</strong>: {{Note}}</p>
```

If the “Note” placeholder had no content, the string “Note:” would appear on the page without anything following it. This would appear sloppy, and could be avoided by enclosing the line in a `vtIf` block:

```
<!--{{vtIf: {Note} .neq. }}-->
    <p><strong>Note</strong>: {{Note}}</p>
<!--{{vtEndIf}}-->
```

In this case, the paragraph with the note would only be displayed if the `{{Note}}` placeholder had a value.

Single Alternatives

To display an alternative text when the placeholder is empty, you should add a `vtElse` directive:

```
<!--{{vtIf: {Note} .neq. }}-->
    <p><strong>Note</strong>: {{Note}}</p>
<!--{{vtElse}}-->
    <i> No note available</i>
<!--{{vtEndIf}}-->
```

Multiple Alternatives

You can also evaluate several conditions, one after the other:

```
<!--{{vtIf: Condition 1 }}-->
    ...
<!--{{vtElseIf: Condition 2 }}-->
    ...
<!--{{vtElseIf: Condition 3 }}-->
    ...
<!--{{vtElse}}-->
    ...
<!--{{vtEndIf}}-->
```

Expressing Conditions

The `vtIf` and `vtElseIf` directives contain an additional specification (condition) after the colon that describes in which situations the following HTML block should appear. This condition is always expressed using the following pattern:

left comparand . relational operator . right comparand

In the above example, the `{Note}` placeholder is the left comparand, the relational operator is `.neq.` (meaning *not equal to*), and the right comparand is null. Be sure to surround the placeholder only with single quotes, because it is used within a VirthosTalk directive.

A placeholder can also be used as the right comparand. In that case, you must be sure to place at least one space between the closing brace of the placeholder and the closing brace of the `vtIf` directive:

```
<!--{{vtIf: {ReminderDate} .eq. {vtCurrentDate} }}-->
    <p><strong>{{ReminderText}}!</strong></p>
<!--{{vtEndIf}}-->
```

Relational Operators

The following table shows the relational operators that can be used in a `vtIf` condition:

Operator	Explanation
<code>.eq.</code>	is equal to
<code>.neq.</code>	is not equal to
<code>.gt.</code>	(number or date) is greater than
<code>.gte.</code>	(number or date) is greater than or equal to
<code>.lt.</code>	(number or date) is less than
<code>.lte.</code>	(number or date) is less than or equal to
<code>.bw.</code>	(string) begins with
<code>.ew.</code>	(string) ends with
<code>.cn.</code>	(string) contains
<code>.ncn.</code>	(string) does not contain

If you are comparing one date with another, they must both be in “dd.mm.yyyy” format. Time expressions must be in “hh:mm” format, where hour and minute values less than 10 must contain a leading zero. However, no leading zeroes are needed in the day and month:

```
<!--{{vtIf: {vtCurrentDate} .eq. 1.1.2007 }}-->
    <p>Happy New Year!</p>
<!--{{vtEndIf}}-->
```

When comparing strings, Virthos ignores case differences. A condition such as `{Name}.eq.jones` would be true whether the `{Name}` placeholder contained the word “Jones” or “JONES.”

Hyperlinks

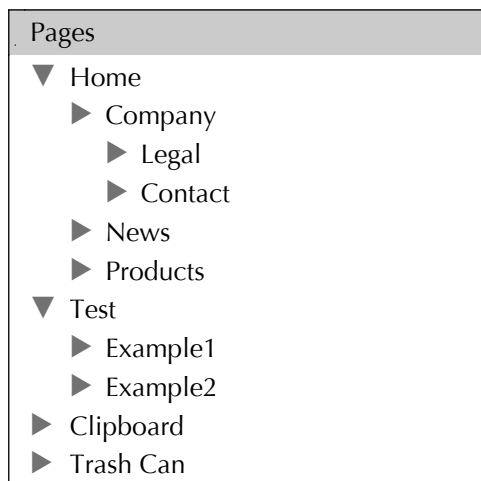
If you want a template to include a hyperlink to a particular Virthos page, the `vtLink` placeholder is available. This is replaced by a URL that points to a particular Virthos page. The exact page is specified by a parameter that can be built in various ways.

Hyperlinks Using Page Names

A typical way of using `vtLink` would look like this:

```
<a href="{{vtLink:/Products}}">Products</a>
```

In this example, the `vtLink` placeholder would be replaced by a URL pointing to a page named "Products." The leading slash indicates that the page is located underneath the designated start page in the Virthos hierarchy:



A link to the "Example2" page could look like this:

```
{{vtLink://Test/Example2}}
```

The double slashes at the beginning in this case indicate that the page specification starts at the top level of the hierarchy.

Hyperlinks Using Page Numbers

Instead of using page names, you can also create hyperlinks using page numbers. Every page that you create with Virthos is assigned a unique number. You can determine the number by placing your mouse over the page name in the Navigation Bar and waiting a moment. The number is then shown under the mouse pointer. A URL pointing to page 132 would look like this:

```
{{vtLink:132}}
```

Hyperlinks Using Relative Pages

There are also various ways of defining a hyperlink *relative to the current page*. For example,

```
{{vtLink:..}}
```

would create a hyperlink to the page subordinate to the current page. And

```
{{vtLink:#3}}
```

would result in a hyperlink to the third subordinate page (the page that clicking on [Structure] shows as being in third position).

Hyperlinks to Template Variants

If you make use of template variants, you can also use *vtLink* to refer to a particular variant. Thus,

```
{{vtLink: -met="print"}}
```

would bring up the URL of the page made available by applying the “print” method to the current page. If you want to point to another page *and* a special method at the same time, you can do that with the help of the `-pg` parameter:

```
{{vtLink: -pg="/Products", -met="print"}}
```

When using *vtLink*, it is often helpful to use single quotes instead of double quotes, since many HTML editors may confuse the latter with the double quotes used in the `<a>` tag.

```
<a href="{{vtLink: -pg='/Products', -met='print'}}">
```

More information on this subject can be found in the chapter “Entry Addresses” in Part 4 of this manual, as well as in the VirthosTalk Reference manual.

Hyperlinks to Other Languages

If you are using Virthos Pro to manage different language versions, you can use *vtLink* to create a link that will direct the user to a particular language. Here is an example:

```
{{vtLink: -lg="de"}}
```

This creates a link that will show the visitor the current page in German. At the same time, it makes German the current language—meaning that all further pages that he accesses will be in German. The current language will remain in effect until the visitor clicks a hyperlink that uses the `-lg` parameter to redirect him to another language.

More information on this topic is available in the topic “Translating Page Content” in the chapter “Managing Pages,” as well as in the chapter “Entry Addresses.”

Changing Context

Most VirthosTalk placeholders and directives require a context to determine how they should be interpreted. The current page—the one currently being displayed in the browser—usually provides this context. This means that a placeholder like

```
{{Headline}}
```

will be replaced by the text that was entered as “headline” for the *current* page, and that

```
{{vtName}}
```

is evaluated as the internal name of the *current* page.

Changing Context with vtLoop

VirthosTalk lets you change the context within a template as often as you like and thus insert content from another page on the current page. An example of changing context is the `vtLoop` directive (see the topic “Dynamic Lists”): This directive repeats an HTML block for each of the current page's subordinate pages. With each repetition, the context shifts to another page. On the first repetition, the `{{Headline}}` would be replaced by the first subpage's headline, and on the second repetition, by the second subpage's, and so on.

Changing Context with vtUse

You can also set the context to another particular page without using `vtLoop`. This can be done with the `vtUse` directive, like this:

```
<!--{{vtUse:/Contact}}-->
  <p>Phone: {{PhoneNumber}}</p>
<!--{{vtEndUse}}-->
```

In this example the `{{PhoneNumber}}` placeholder would be replaced by the text that was entered as the phone number whenever the “Contact” page was created or edited—without regard to which page is currently being viewed.

The page specification following the colon in the `vtUser` directive is formed under the same rules as a page in `vtLink` (see the topic “Hyperlinks”). The `vtUse` directive is especially useful when building dynamic navigation menus.

Repetition

The topic “Dynamic Lists” has already shown one way of repeating an HTML block several times on a page. VirthosTalk offers a second possibility that, although it does not permit complex structured lists, is easy to implement. Suppose you want to build a selection list like this:

```
<select name="weekdays">
  <option>Mon</option>
  <option>Tue</option>
  <option>Wed</option>
  <option>Thu</option>
  <option>Fri</option>
</select>
```

You can also use VirthosTalk to build the same selection list for you:

```
<select name="weekdays">
  <!--{{vtRepeat: Mon, Tue, Wed, Thu, Fri}}>
  <option>{{vtCurrentItem}}</option>
  <!--{{vtEndRepeat}}-->
</select>
```

The `vtRepeat` directive causes the following HTML block (up to the ending `vtEndRepeat`) to be repeated once for each item in the list that was named in the parameter list. Within each `vtRepeat` block, the `vtCurrentItem` placeholder can be used to reference the current list item.

The list to be processed with `vtRepeat` can also be specified with a placeholder, for example:

```
<ul>
<!--{{vtRepeat: {References} }}-->
  <!--{{vtUse: {vtCurrentItem} }}-->
    <li><a href="{{vtLink}}">{{vtName}}</a></li>
  <!--{{vtEndUse}}-->
<!--{{vtEndRepeat}}-->
</ul>
```

This creates a list of hyperlinks. The placeholder “References” must contain a list of page numbers separated from each other either by commas or line breaks (returns).

Variables

If you need to, you can refer to values that belong to another context from outside of that context. For this purpose, VirthosTalk lets you store the content of placeholders in variables that can be accessed later. So, in this directive:

```
<!--{{vtSet: Current_Page="{vtID}" }}-->
```

the number of the current page is stored in a variable called "Current_Page." The placeholder

```
{{vtGet: Current_Page}}
```

lets you access the content of this variable in another place that does not depend on the context in which it was saved.

Within a `vtSet` directive, you can set several variables at the same time:

```
<!--{{vtSet: Name="{vtName}", Number="{vtID}", Title="{Title}"}}-->
```

You can also specify the value of a variable using several placeholders and/or literals:

```
<!--{{vtSet: Full_Name="Page_{vtName}_{vtID}" }}-->
```

You can assign any name that you wish to variables; however, you must respect the same rules as for naming user placeholders (see the topic "Editable Text"). You should also be aware that the value of a variable is only available for one invocation. This means that you cannot use `vtGet` to retrieve a value that was set by a previous invocation of a page with `vtSet`.

Library Files

Even if you create a large number of templates for a website, there is often little difference between them. Large sections of HTML code are often identical, because the basic layout and navigation elements are usually the same from one page to another. To make it easier to maintain a large number of similar templates, you can store often repeated blocks of code in their own files. When you change them later, the results are immediately effective within all templates that include these “library files.”

To integrate the content of a library file in a template, you can use the `vtInclude` placeholder in the following way:

```
{{vtInclude: includes/header.html}}
```

After the colon comes the path to the library file, relative to the template's location. In this example, there must be a subdirectory named “includes” under the directory in which the template file is located, and that subdirectory must include a library file named “header.html.”

It is always a good idea to place library files in subdirectories, so that they will not be confused with the template files. You should keep in mind that relative path specifications, as for image or style sheet files, are always interpreted relative to the location of the template, not from the location of the library file. Another way to avoid confusing library files with templates is to give the library files their own file extension, such as “.inc” or “.incl.”

The same rules as for creating templates apply to the creation of library files. They can contain all available VirthosTalk placeholders and directives, including other `vtInclude` placeholders.

Placeholders in Library Files

If a template's context is changed before a `vtInclude` placeholder is referenced, any VirthosTalk placeholders and directives in the library file will be interpreted within the changed context. The reverse is also true: If the context is changed within a library file, the changed context will subsequently be in effect in the template, as well.

Values set by `vtSet` in a template can be retrieved by using `vtGet` in a library file, and vice versa. This makes it possible for a library file to be used with a degree of flexibility. If you want to move the complete HTML header area to a library file, for example, you face the problem that the header usually includes the page title, which would probably vary from template to template. The solution is to define the title in the library file using the `vtGet` placeholder:

```
<title>{{vtGet:PageTitle}}</title>
```

and to store the page title in the template using a `vtSet` directive *before* the `vtInclude` placeholder:

```
<!--{{vtSet: PageTitle="Smith & Jones Inc.: ContactPage"}}-->
{{vtInclude: includes/header.html}}
```

In this way, you can include the HTML header and still allow each template to use a different page title.

Please note!

If you use a web design program like GoLive or Dreamweaver to design templates, Virthos library files have only limited usefulness. These programs usually assume that you are designing complete HTML pages and not just parts of an HTML page. On the other hand, they also have their own functions for simplifying the editing of HTML pages with the same basic structure. ¶

URL Parameters

Sometimes it is desirable to use the same template in quite different circumstances. In order for the template to be tailored to the current situation, you need to supply additional information when invoking the page and to evaluate this information within the template. This can be accomplished by using URL parameters and the `vtGetValue` placeholder.

A URL parameter is a part of a URL used to pass a named value. For example, the URL

```
http://.../virthos.php?-pg=123&-met=print
```

contains two parameters: the parameter `-pg` with the value “123” and the parameter `-met` with the value “print.”

A URL parameter is composed of the parameter name, an equals sign and the value. If a URL contains multiple parameters, these must be separated from each other by an ampersand (“&”). The first URL parameter is always preceded by a question mark. More information on this topic is available in the chapter “Entry Addresses” in Part 4 of this manual).

Virthos makes intensive use of URL parameters, but also allows template designers to use URL parameters for their own purposes. To pass a value when invoking a page, you only need to define it as a parameter in `vtLink`:

```
{{vtLink: -pg='/Products', showAll='yes'}}
```

This example shows a URL that contains not only Virthos parameters but also the URL parameter “showAll” with a value of “yes.” In the template on which the “Products” page is based, you could use `vtGetValue` to check whether this parameter has been set, and take the proper action:

```
<!--{{vtIf: {vtGetValue:showAll} .eq. yes }}-->
```

Here, you have the HTML code that should be included if the URL parameter “showAll” has the value “yes.”

```
<!--{{vtElse}}-->
```

Here, you have the HTML code that should be included if the URL parameter “showAll” has not been set or has a value other than “yes.”

```
<!--{{vtEndIf}}-->
```

There are two points to remember when naming URL parameters:

- The name can consist only of characters that are allowed in URL's. These are generally the letters of the alphabet, the hyphen, underscore, and the digits 0 through 9.
- The name may not begin with a hyphen or minus sign, since this initial character is reserved for Virthos' own URL parameters.

Form Data

Besides URL parameters, there is a second way to pass values from one template to another: through HTML forms. A typical example of using a form would be:

```
<form action="{vtLink:/Products}" method="post">
  <input type="text" name="keyword" size="30">
  <input type="submit" value="go">
</form>
```

The first line defines the form action, which is a hyperlink to a particular Virthos page. It is important to use the “post” form method. The second type of method recognized by the HTML standard—“get”—should not be used within Virthos.

In the second line, there is an entry field with the name “keyword.” This is where your visitor can enter text later. The third line generates a button marked “go”; when the visitor clicks it, the form is submitted and the target page defined by the action is invoked.

In the template on which the target page is based, you can use the vtPostValue placeholder to query the value of the form field.

```
<!--{{vtIf: {vtPostValue:keyword} .eq. software }}-->
```

Here, you have the HTML code that should be included if the form field “keyword” contains the value “software.”

```
<!--{{vtElseIf: {vtPostValue:keyword} .eq. hardware }}-->
```

Here, you have the HTML code that should be included if the form field “keyword” contains the value “hardware.”

```
<!--{{vtElse}}-->
```

Here, you have the HTML code that should be included if the form field “keyword” contains some other value or if the page has been accessed directly (not through a form).

```
<!--{{vtEndIf}}-->
```

The names of form fields can be just about anything you wish, with the same restrictions as for URL parameters: no accented letters, special characters, etc., and the first character cannot be a hyphen or minus sign, since that is reserved for Virthos' internal names.

Processing Forms

Virthos offers a variety of ways to process HTML forms. These forms can be part of an ordinary template, and can serve purposes as varied as automatic sending of emails or modifying Virthos content. This chapter describes the basics of how Virthos processes forms, and how can you use these in your own projects.

Basics of Working With Forms

The topic “Form Data” in the previous chapter already illustrated one example of how you can use forms in Virthos. There, we discussed how values can be passed from one page to another—but there was no real “processing” of the form data taking place. To tell Virthos how to process the information from a form, you must connect the information to a Virthos action (not to be confused with the form action). The first line of the HTML form would have to look something like this:

```
<form action="{vtLink:-act='ActionName'}" method="post">
...
</form>
```

The “ActionName” parameter tells Virthos what to do with the form data. If the data should be sent in an email, the name of the action would be “mail.” Some examples of actions are presented in the following topics; for a full discussion, please refer to the VirthosTalk Reference Manual.

The `vtLink` placeholder in the above example could carry with it other parameters besides the action name—for example, `-pg` to invoke a particular page or `-met` to invoke a particular method. This does not affect how the action itself is carried out. For some actions, additional parameters must be specified, since otherwise, the action cannot be executed. The following topics give more information about this, as well.

Results of Form Processing

You can determine whether a form action has been successfully executed with the help of the VirthosTalk placeholder `{vtResult}`. If this placeholder is empty or it contains a positive number, the action was successful. If it contains a negative value, there was an error in processing the form data. You can use a corresponding `vtIf` block to catch the error and possibly issue an error message.

Form Elements

When designing an HTML form, please pay close attention to the names of the individual elements, whether they be text elements, list boxes, or something else. The name of the element may not contain spaces, special characters (except for underscores), or accented letters. The first character must be alphabetic; digits may follow after that. Most of the time, uppercase and lowercase characters in Virthos are considered to be the same. Still, it is a good idea to have a uniform naming convention within a project.

Virthos process basically all the elements that HTML allows in forms. The use of some elements may affect how forms are automatically processed. The following topics will describe these points.

Check Boxes

An example: `<input type="checkbox" name="active" value="1">`

Most browsers transmit an `<input>` check box in a form only if the user has selected, or checked it. To ensure that an element is transmitted even if it is empty, or not checked, you should declare a hidden element of the same name before the actual check box, but without giving it a value. In the above example, this would be:

```
<input type="hidden" name="active" value="">
```

If the field is actually selected, Virthos is sent two values for the same element (a null value and the value "1"). In this case, the last transmitted value always takes precedence. That is why the hidden form element needs to come *before* the check box element.

Submit Buttons

An example: `<input type="submit" value="proceed">`

The button that submits the form can also carry content with it. If you do not want to do this, do not give the button a `name` attribute. You can still place a description on the button with the `value` attribute.

File Selection Lists

An example: `<input type="file" name="MyFile">`

If you want to allow a form to permit a file to be uploaded, you must use the `enctype="multipart/form-data"` attribute for the `<form>` element. Otherwise, Virthos will not be able to process the uploaded file. The first line of the form could look like this:

```
<form action="{vtLink:-met='thanks',-act='mail'}" method="post" ␣
                                enctype="multipart/form-data">
```

Multiline Entry Fields

An example: `<textarea name="Tip" rows="4" cols="50">{{Tip:raw}}</textarea>`

If a multiple-line entry field is to be assigned to a VirthosTalk user placeholder as soon as the form is loaded, the placeholder must include the `raw` parameter, as in the above example. Without this specification, each line break in the text being entered would cause a `
` tag to be generated.

List Boxes

An example: `<select name="weekday[]" size="3" multiple>`

If a list box allows the user to select *more than one* item, the name of the `select` element must end with brackets, as in the above example. Otherwise, Virthos would process only the last of the selected elements. Please note, however, that the use of brackets could make it difficult to use the name in JavaScript statements.

Sending Form Data in Emails

This may be one of the most common types of form action. It transmits all the data that a user has entered on a form to a designated email address. You enable this by setting the `-act` parameter to the value "mail." No other parameters are required. You can specify a subject for the email by including the `-subject` parameter, though.

The "mail" action refers to the "Email" settings on the Settings page (see the chapter "Managing System Settings" in Part 2 of this manual). The addresses that you have specified are used to determine the sender and addressee. For security reasons, you should not specify these addresses in the email form itself. That could cause spam robots to read them and use them to generate spam.

The message content is assembled according to the following pattern:

```
Fieldname1: Fieldcontent1
```

```
Fieldname2: Fieldcontent2
```

```
...
```

If the form looks like this:

```
<form action="{vtLink:-act='mail'}" method="post">
  <p>Name: <input type="text" name="Name"></p>
  <p>Phone: <input type="text" name="Phone"></p>
  <p><input type="checkbox" name="Callback" value="yes">
    please give me a call</p>
  <input type="submit" value="Submit">
</form>
```

the resulting email message could look like this:

```

This message was automatically created by Virthos.
Path to Script: www.mycompany.com/virthos/virthos.php

Name: John Smith

Phone: (555) 568-1234

Callback: yes

```

The “mail” action also processes files uploaded with an `<input>` form element of type `file`. These files are automatically sent as email attachments, as long as their length does not exceed the maximum size specified on the “Email” page (see the chapter “Managing System Settings” in Part 2 of this manual). Please note that in order to process uploaded files, the `<form>` tag must have its `enctype` attribute set to `multipart/form-data`, as described in “File Selection Lists” under the topic “Basics of Working With Forms.”

A User-Designed Login Form

In order to create your own form to replace the Virthos-supplied login window, its action name must be set to “login.” You also need to provide a form field named “username” and one with the name “password.”

```

<form action="{vtLink:-act='login'}" method="post">
  <p>Name: <input type="text" name="username"></p>
  <p>Password: <input type="password" name="password"></p>
  <p><input type="submit" value="Log In"></p>
</form>

```

In Virthos Basic, you can leave out the “username” field or disable it using the `disabled` attribute.

Please take note that when preparing your own login form, if users encounter any errors while logging in, they will automatically be redirected to the “vtlogin” method. If your login form does not already use this method name, you should also design your own method for this template, preferably in the form of a global method (see the chapter “Template Variants”). Otherwise, after an error, your users would see Virthos’ login window.

Creating New Pages Using a Form

If the form information is to be used to create a new Virthos page, the action name must be “create.” As an additional parameter, you must supply the name of the template that the new page should be based on, for example:

```
<form action="{vtLink:-act='create',-template='posting'}" ↵
                                     method="post">
  <p>Subject:<br> <input type="text" name="subject"></p>
  <p>Content:<br> <textarea name="content"></textarea></p>
  <p><input type="submit" value="Submit posting"></p>
</form>
```

If the new page is not to be a subpage of the current page, you must supply `-target` as an additional parameter to specify the target page. This is done in the same way as when using the `-pg` parameter with the `vtLink` placeholder.

The names of the individual form elements (text fields, list boxes, etc.) must correspond to the user placeholders in the specified template. Virthos will still save the form fields even if they do not match the placeholder names; however, the contents will not be visible on the new page.

When using list boxes that permit more than one selection, the name of the select element should end with a set of brackets, as previously described in the topic “Basics of Form Processing.” The same applies when several check boxes relate to the same placeholder, for example:

```
<input type="checkbox" name="weekdays[ ]" value="Mon"> Mon
<input type="checkbox" name="weekdays[ ]" value="Tue"> Tue
<input type="checkbox" name="weekdays[ ]" value="Wed"> Wed
...
```

If a user were to check both of the first two options, the placeholder `{{weekdays}}` would then cause two lines to be displayed:

```
Monday
Tuesday
```

If the template referenced by the “create” form contains placeholders of types `vtMedia` or `vtFile`, you can create form elements for uploading files in the following manner:

```
<input type="file" name="Image01">
```

In this case, unlike other form elements, it is absolutely mandatory for the underlying template to actually have a placeholder with this name. It is also essential that the `<form>` element have the `enctype="multipart/form-data"` attribute, as described in the topic “Basics of Form Processing.”

As with all actions, the “create” action uses the `-pg` parameter in `vtLink` to determine which page to display after the form is submitted. This parameter can only be used with pages that already exist. If

you want to access the newly created page (which does not exist yet when the form is called and whose page number is thus not yet known), you must use the special `-check` parameter:

```
{{vtLink:-act='create',-template='posting',-check='1'}}
```

The `-check` parameter causes Virthos to direct the user's web browser to the new page as soon as the form information is processed, so that you can check that page immediately. When `-check` is user, Virthos ignores any `-pg` parameter.

If a customized “create” form is automatically being used, when an author selects a page type for creating a new page, you must make the form available as a template variant with the “vtnew” method. For example, if you design a form for creating pages of type “posting,” the template must have been saved under the name `posting.vtnew.html`.

Changing Page Content with a Form

If a form is intended to update the contents of an existing page, the action name must be “update.” The number of the page to be updated must be supplied with the `-obj` parameter. Since most of time, it is the current page that is involved, the value can be supplied with the `vtID` placeholder:

```
<form action="{{vtLink:-act='update',-obj="{{vtID}}}" method="post">
  <p>Subject:<br>
  <input type="text" name="subject" value="{{Subject}}"></p>
  <p>Content:<br>
  <textarea name="content">{{Content:raw}}</textarea></p>
  <p><input type="submit" value="Save"></p>
</form>
```

The same rules are in effect for naming form elements as with the “create” action (see previous topic). A special feature of the “update” action is that the form elements are generally initialized to the current page contents. The above example shows how this might look for single-line and multiple-line text fields. For check boxes, the initialization could look like this:

```
<input type="hidden" name="weekdays" value="">

<input type="checkbox" name="weekdays[]" value="Monday" ↵
  {{vtIf:{weekday}.cn.Monday}}checked{{vtEndIf}}>

<input type="checkbox" name="weekdays[]" value="Tuesday" ↵
  {{vtIf:{weekday}.cn.Tuesday}}checked{{vtEndIf}}>

...
```

Note the hidden field that is declared first. If none of the check boxes have been selected (or if they have all been cleared), any existing content in the “weekdays” placeholder will be deleted (see the topic “Basics of Working With Forms”).

You can initialize a list box in the following manner:

```
<select name="weekdays[]" size="5" multiple>
  <option {{vtIf:{weekday}.cn.Monday}}selected{{vtEndIf}}>
    Monday
  </option>
  <option {{vtIf:{weekday}.cn.Tuesday}}selected{{vtEndIf}}>
    Tuesday
  </option>
  ...
</select>
```

If the template referenced by the “update” form contains placeholders of types `vtMedia` or `vtFile`, you can create objects on the form for uploading of files. When a file is uploaded, it replaces any file that was previously assigned to the placeholder.

To give authors the ability to delete a previously uploaded file without replacing it with a new one, you can make use of another special form element:

```
<p>New Image: <input type="file" name="Image01"></p>
<p><input type="checkbox" name="mediaMethodImage01" value="remove">
Remove image</p>
```

The name of the additional form element is built by prefacing the name of the `vtMedia` placeholder with word “mediaMethod.” A similar form element for a `vtFile` placeholder is built by placing “fileMethod” before the placeholder name:

```
<p>New File: <input type="file" name="File01"></p>
<p><input type="checkbox" name="fileMethodFile01" value="remove">
Remove file</p>
```

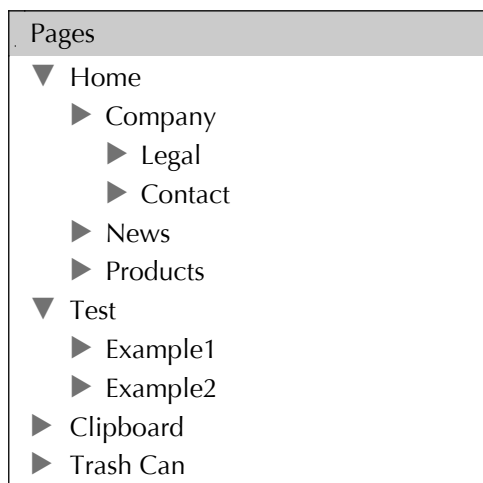

Dynamic Navigation Menus

When you are responsible for a complex website, you may often not know in advance exactly which web pages you will actually publish. That is why Virthos lets authors create new pages when appropriate. So that these new pages can become accessible to the public, the use of “dynamic” navigation menus is required. We use the term “dynamic” because these menus automatically expand and contract when new pages are created or existing pages are deleted.

Dynamic navigation menus in Virthos, just like lists, are built using the `vtLoop` directive. The use of `vtLoop` means that an HTML block is repeated for each page that is subordinate to the current page. A simple example of a dynamic navigation menu might look like this:

```
<ul>
<!--{{vtLoop}}-->
    <li><a href="{{vtLink}}">{{vtName}}</a></li>
<!--{{vtEndLoop}}-->
</ul>
```

This creates a list with a hyperlink to access each subordinate page. Assuming the following page structure:



the list on the “Home” page would look like this:

- [Company](#)
- [News](#)
- [Products](#)

and the one on the “Company” page, like this:

- [Legal](#)
- [Contact](#)

In order to make the main level of navigation available on the “Company” page (that is, the list of pages subordinate to “Home”), the `vtLoop` block must be enclosed in a `vtUse` block:

```
<ul>
<!--{{vtUse://Home}}-->
  <!--{{vtLoop}}-->
    <li><a href="{{vtLink}}">{{vtName}}</a></li>
  <!--{{vtEndLoop}}-->
<!--{{vtEndUse}}-->
</ul>
```

As we mentioned in the “Introduction to VirthosTalk,” the `vtUse` directive changes the context for the interpretation of the subsequent VirthosTalk placeholders and directives. In the above example, the context is first shifted to the “Home” page. As a result, the following `vtLoop` directive causes the page under the “Home” page to be listed, no matter which page is currently being viewed.

Since `vtLoop` directives can be nested to as many levels as needed, you can use this method to build multilevel navigation menus. To create two levels of navigation, you could build on the above example:

```
<ul>
<!--{{vtUse://Home}}-->
  <!--{{vtLoop}}-->
    <li><a href="{{vtLink}}">{{vtName}}</a></li>
    <ul>
      <!--{{vtLoop}}-->
        <li><a href="{{vtLink}}">{{vtName}}</a></li>
      <!--{{vtEndLoop}}-->
    </ul>
  <!--{{vtEndLoop}}-->
<!--{{vtEndUse}}-->
</ul>
```

Starting from the above page structure, and assuming that the “News” and “Products” pages have no subpages, the result would look like this:

- [Company](#)
 - [Legal](#)
 - [Contact](#)
- [News](#)
- [Products](#)

The possibilities for refining dynamic navigation menus are almost endless. By using `vtIf` directives, you can “unlock” only the menu items to which the current page belongs, or you can visually highlight the current page within the menu.

Troubleshooting

Giving a non-compliant name to a template file or misspelling the name of a VirthosTalk placeholder or directive are two common errors that can cause a template not to function correctly. The following table lists some commonly occurring problems and offers suggestions for correcting them:

<i>Problem</i>	<i>Possible Cause</i>	<i>Solution</i>
A particular template does not appear in the list of templates when creating a new page.	The template was not copied into the correct directory.	Place the template directly in Virthos' "templates" directory or in a subdirectory.
	The template file's extension is not ".html."	Rename the template file so that it ends in ".html."
	The template belongs to a different template package from the current page's.	Click "Show all packages" in the template list.
The resources used by the template, such as images and style sheets, do not appear in the browser.	The resources are located in the same directory as the template, or a higher directory or one on the same level.	Place the resources in a subdirectory of the one the template is in, and ensure that the path in the template is correctly specified.
While trying to access a page, the browser displays an error message beginning with "Parse error."	There is a block directive like <code>vtIf</code> or <code>vtLoop</code> in the template without a matching ending directive.	Verify that each <code>vtIf</code> and <code>vtLoop</code> directive (or similar ones) has a matching <code>vtEndIf</code> or <code>vtEndLoop</code> directive, and provide these, if missing.

PART 4

Integration

Entry Addresses

If you want to create a link from a page on an existing website to a page within Virthos, when calling the *virthos.php* script, you must include some identifying information about the page to be displayed. Without this information, the script would always bring up the default start page. If you want to point to a different page, there are two possibilities:

- specifying the page number, or
- specifying the path.

This chapter describes how to use both of these methods and also how to invoke template variants and languages.

General Observations About Address Parameters

The page to be displayed by the *virthos.php* script and the applicable method and language can be specified by an address parameter (also known as a URL parameter). To clarify where the address itself ends and where the parameters begin, use a question mark as a separator:

Address?AddressParameter

An example might look like this:

`www.mycompany.com/virthos/virthos.php?login`

Here, the *virthos.php* file, located in the *virthos* directory on the *www.mycompany.com* server, is being invoked with the “login” parameter.

If you want to specify multiple parameters, separate each of them with an ampersand (&):

Address?Parameter1&Parameter2&Parameter3

Generally, when you do this, you will also want to switch from positional to keyword parameters, meaning that each parameter name is followed by a parameter value, with an equals sign between them:

`www.mycompany.com/virthos/virthos.php?-pg=162&-met=print`

Here, the *virthos.php* file is being called with two parameters: the parameter `-pg` and the parameter `-met`. The `-pg` parameter has the value “162” assigned to it, and the `-met` parameter is assigned the value “print.” Precisely how this takes place is described in the following topics.

Please note!

In the above example, the parameters are reproduced as you would enter them in the address field of a web browser. If you specify an address like this within an HTML page, however, as in the `href` attribute of an anchor, you must replace the ampersand with the string `&`. Most web browsers will be able to interpret even the short form, but this usage does not conform to the HTML standard, and can lead to complications especially where accessible websites are concerned. ¶

References by Page Number

Within Virthos, each page has a unique number that is assigned to it when it is created. This number remains associated with it as long as the page exists. This is true even when the page is moved or otherwise altered. The number lets you refer to a page concisely and unambiguously. For example, the reference

```
<a href="virthos/virthos.php?381">...</a>
```

always points to Virthos page 381, no matter where it is located in the Virthos page hierarchy. (The path indication before `virthos.php` depends on where exactly the Virthos subdirectory is located on the web server. (This particular path is only one possibility, as it is also in the example below.)

There are several ways to find out the number of a Virthos page:

- Move your mouse over the page in Virthos Manager's Navigation Bar, and wait a moment. The page number will appear underneath the mouse pointer as a tool tip.
- If the page is currently shown in Virthos Manager, move the mouse pointer over the file symbol to the left of the page name in the Header Area. After a moment, the page number will appear as a tool tip.
- Use Virthos Manager to navigate to the page; then click [Specifications] in the Mode Selector. In the window that appears, you will find the page number under the page name and the template identifier.

References by Path

References by page number are short and unambiguous, yet abstract: After a few days, who can remember what page 381 is? Virthos therefore offers a second, more descriptive way of specifying pages—through a path. The path includes the page name and possibly an indication of where the page resides in the Virthos page hierarchy. This method is similar to the way files are referenced on a hard drive, or the pages of an existing website: You list the names of all the pages that a user must pass through in order to arrive at the page concerned:

```
<a href="virthos/virthos.php?/Products/Computers/Laptops">...</a>
```

This example is a reference to the “Laptops” page under the “Computers” page, which, in turn, is under the “Products” page. The “Products” page is under the designated start page.

To refer to a page that is not directly under the start page, you must precede the path specification with a double slash:

```
<a href="virthos/virthos.php?//Downloads">...</a>
```

This hyperlink refers to a page named “Downloads,” which is at the highest level of the Virthos hierarchy. It can be on a branch parallel to the designated start page, or it can be the start page itself.

Please note!

Only certain letters, digits and characters are allowed in references. If a page in a path specification includes spaces or special characters, these must be coded in a special way—spaces become %20, for example. To see which characters are affected and what codes to use in their place, you can consult any of several online references, such as www.htmlhelp.com. ¶

A shortcut to using a path specification in a reference is:

1. Access the desired page in Virthos Manager.
2. Click the folder symbol to the left of the path specification in Virthos Manager's Header Area. The current page is then opened in a new browser window.
3. Copy the contents of your browser's address field to create the reference.

In contrast to references by page number, a path specification can become invalid if you move or rename pages within Virthos. Therefore, when it comes to pages that you might use as entry addresses, you should protect them against renaming and moving (see the topic “Modifying Page Specifications” in the chapter “Managing Pages”). On the other hand, path specifications are more flexible, and they remain valid when you replace a page with another one having the same name. This can often be helpful when revising a website. Path specifications are particularly helpful for referring to pages at the top levels of the hierarchy.

Accessing Template Variants

If you want to create a reference to a page displayed with a particular template variant (“method”), you can do this using an additional URL parameter:

```
<a href="virthos/virthos.php?-pg=381&-met=print">...</a>
```

Since two parameters are needed in this case (the page and the method), you must use keyword parameters. The page is specified with the `-pg` parameter (for “page”), and the method with the `-met` parameter. The two parameters are separated from one another by an ampersand (&).

Accessing Language Versions

If using Virthos Pro for a multilingual website, you can also specify the `-lg` parameter when you invoke the *virthos.php* script, and thus designate the language in which the page should be displayed.

```
<a href="virthos/virthos.php?-pg=381&-lg=de">...</a>
```

In this example, the German version (“de”) of page 381 is invoked. Just as when you invoke a particular method (see previous topic), you must also use a keyword parameter (`-lg`) in this case.

You can specify any value listed in your System Settings under “Languages” as a value for the `-lg` parameter. If by chance there is no version of the page in that language, Virthos reverts to the language found in the top position under System Settings. If there is no content in this language, Virthos goes to the language in second position, and so on.

Please note!

When a user is directed to particular language version by the `-lg` parameter, this language remains active for him until he is explicitly redirected to another language. ¶

Invalid References

When invoking the *virthos.php* script with a page number or path specification, it is possible that the requested page either does not exist or cannot be found. This may mean that you have entered the wrong page number or path, or the page has been deleted or renamed. When this happens, Virthos does not display an error message, but takes you to its designated start page instead. In the rare case where even this is not possible, because the start page no longer exists (or is no longer publicly accessible because access rights have been set to block it), Virthos displays an internal start page that contains the site name and logo, as well as links to all the pages at the top level of Virthos' hierarchy.

If you specify a non-existent method, Virthos responds in a similar fashion: If someone clicks a link pointing to a non-existent method, *no* error message is displayed. Instead, the default method is used.

This does not apply to attempts to access pages within Virthos Manager: When logged in, if you click on a link to an invalid page or method, you will see a succinct error message that will let you quickly notice and correct the error.

Virthos as a Start Page

Using the techniques described in the previous chapter, it is a simple matter to direct someone viewing an existing web page to a Virthos page. If necessary, you can also allow Virthos to act as your website's start page (the page that you would see by entering `www.mycompany.com`). This chapter describes the different ways that you can accomplish this, and some practical ways to make these work for you.

Frame-Based Layouts

This approach uses HTML's frame concept, which allows several pages to be combined in one window. The actual start page—(*index.html*) in this case—defines a frameset containing a single frame that fills the entire window, and whose source is the *virthos.php* file:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
<html>
<head>
  <title>Frame-Test</title>
</head>
<frameset>
  <frame src="http://www.meinefirma.de/virthos/virthos.php">
</frameset>
</html>
```

The advantage of this approach is that a frame layout is very easy to implement and works on any web server and on all of today's browsers. The disadvantage is that the start page's URL remains displayed in the browser's address field, no matter what page the visitor has actually accessed. This can cause many browser functions, including the setting of bookmarks and the printing of pages, to sometimes produce unintended results.

Automatic Redirection

You can also set up your start page to point to Virthos by providing your actual start page (*index.html*) with a meta tag that automatically redirects the browser to the *virthos.php* script.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
  <title>Meta-Test</title>
  <meta http-equiv="refresh" content="0;
    URL=http://www.mycompany.com/virthos/virthos.php">
</head>
<body>
<p>If you do not want to be automatically redirected, just click
  <a href="virthos/virthos.php">here</a></p>
</body>
</html>
```

Redirecting someone through a meta tag is a fairly common practice, but it is not supported by all types of browsers. It can lead to difficulties especially on accessible Internet sites, where users are using special screen reader programs. Therefore, as in the above example, you should also provide a hyperlink that the user can click to move to the Virthos environment.

Reconfiguring a Web Server

The cleanest solution, from a technical viewpoint, to make Virthos your start page is this: Configure your web server so that instead of serving a page named *index.html* (or whatever your start page's name may be), it displays a file named *virthos.php* as soon as a visitor enters the server address. This approach may not be available on all web servers, but when it is, it is preferable to the other methods described above.

Reconfiguration can be accomplished using the *.htaccess* file located in your web server's root directory (the same one where *index.html* is located). This is a simple text file that can be opened and edited with any text editor and that must contain the following line:

```
DirectoryIndex virthos.php
```

Please note that the file name must begin with a period. The result is that the file will not be displayed by the file manager, such as Explorer or Finder, on various operating systems. Therefore, we recommend that you first call the file simply *htaccess* (without the period) and rename it later, after you have uploaded it to the server.

Integrating Virthos With an Existing Website

Not every page of your site needs to be regularly updated. The result is often that a company may decide to use Virthos to deliver certain pages to the browser, but leave the rest as they were previously. In principle, there is nothing wrong with this approach, but you must give some consideration to how the two parts can relate to each other.

- The “Entry Addresses” chapter describes how you can refer to existing pages from inside Virthos. To go in the opposite direction—from Virthos to an existing page—there are no special rules. You do it in the same way that you would normally refer from one existing page to another. Just keep in mind that the browser will interpret any relative paths from the standpoint of the `virthos.php` file, and not from the standpoint of the template concerned.
- If an author clicks a link in Virthos Manager that refers to a web page outside Virthos, that page is displayed In Virthos Manager, but Virthos Manager itself is not updated: The Header Area and Navigation Bar continue to refer to the most recently displayed Virthos page as the current page. That means that if someone clicks [Edit] or [Specifications], the last Virthos page will again be displayed.
- If a website creates and processes session-related data (as in on online store), it is a good idea to avoid references that go outside of Virthos. If a visitor were to click on such a link, his current session would be interrupted, and any data accumulated before that point (such as the contents of a shopping cart) would be lost.

If an Internet site consists only of Virthos pages and a single existing page outside of Virthos—the start page—it is best to make a copy of this page within Virthos, and to refer then only to this internal copy. Both authors and visitors would remain inside the Virthos environment after clicking on this link, and the problems mentioned above would be avoided.